REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE 29JUN2006	3. REPORT TYPE AND DATES COVERED Final 01 Aug 05 - 31 May 06

4. TITLE AND SUBTITLE
A Biologically-plausable Architecture for Shape Recognition

5. FUNDING NUMBERS
W911NF-05-1-0330

6. AUTHOR(S)
Wesley Snyder

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
North Carolina State University

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U. S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

48823.1-MA-II

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

In this project, we explore a new approach to two-dimensional shape recognition. The method draws from literature on the Hough transform and its extensions. The methods is shown to be invariant to zoom, translation, rotation, and partial occlusion, although not zoom and partial occlusion simultaneously. The method is shown to be robust to distortions which smooth the contour shape (scale space changes). Furthermore, when the method misclassifies a shape, it chooses a shape which is most ``similar" (in a human-intuitive sense) to the original.

The method is developed and evaluated on a data base of tank silhouettes and a data base of fish silhouettes.

The computer-based version of the algorithm is shown to have a reasonable implementation in neural hardware, and a neural-network implementation is described.

14. SUBJECT TERMS
Computer Vision, Shape recognition, target classification, automatic target recognition, ATR

15. NUMBER OF PAGES
45

16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION ON THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	**UNCLASSIFIED**	**UNCLASSIFIED**	**UL**

A Biologically-plausable Architecture for Shape Recognition

Wesley Snyder, Ph.D., Senior Member, IEEE
Karthik Krish, Student Member, IEEE
Sanketh Shetty, Student Member, IEEE
Contract Number W911NF-05-1-0330
September 2005 - May, 2006

June 30, 2006

Abstract

In this project, we explore a new approach to two-dimensional shape recognition. The method draws from literature on the Hough transform and its extensions. The methods is shown to be invariant to zoom, translation, rotation, and partial occlusion, although not zoom and partial occlusion simultaneously. The method is shown to be robust to distortions which smooth the contour shape (scale space changes). Furthermore, when the method misclassifies a shape, it chooses a shape which is most "similar" (in a human-intuitive sense) to the original.

The method is developed and evaluated on a data base of tank silhouettes and a data base of fish silhouettes.

The computer-based version of the algorithm is shown to have a reasonable implementation in neural hardware, and a neural-network implementation is described.

Contents

1 Introduction

This work was motivated by a singular experience of one of the authors:

> I was looking over a scene involving a number of man-made objects such as houses, factories, warehouses, and a water tower, as well as some natural objects including a river, trees, grass, etc. In my peripheral vision, I became suddenly aware of a "straightness" which appeared and vanished. In the instant it took my attention to shift, I realized the "straight" thing I had observed was an accidental alignment of the top of a building, the top of a water tower, and a bird which flew between the other two. I had detected a straight "'edge" or "line" (we don't need to distinguish them here), and I had done so without setting the focus of attention (FOA).
>
> We usually think of neural computations done without FOA as being more associated with "hardware" or previously adapted special neural structures, as opposed to neural processes which require FOA, which we associate with "software" or higher -level processing. High level processing is slower, and this feature detection was fast - when my attention shifted, the bird had barely moved. Could a receptive-field-like detector have fired? Unlikely, because the event covered 1/4 of the visual field.
>
> If not a receptive field, based on correlation, then what kind of neural hardware or algorithm could produce such a rapid response? The Hough transform comes to mind. While it is unlikely that trig functions are calculated continuously in the brain, there is evidence that similar calculations do occur in the brain (we discuss literature in section 2).
>
> Once the thought of the Hough transform occurs, one is tempted to ask if structures more general than straight lines can be and are recognized by neural hardware in a similar way - which leads us to the generalized Hough transform.

1.1 Objective

Our objective in this work is to develop a technique for shape recognition which mimics the capabilities of a human while being implementable in hardware of reasonable neural complexity. This mimicking of human capability requires at least some attention to known features of visual processing by humans, including experimental results from neuroanatomy and cognitive psychology. Relevant work in these areas are mentioned in section 2. However, we certainly know that, with 10^{10} neurons and as many as thousands of synaptic connections to each neuron, memory capacity is certainly large in the brain. We furthermore know that computation is relatively cheap - double-precision floating point is not available, but we are allowed a lot of low precision computation.

We thus seek an algorithm and a biologically plausable architecture which will identify individual regions in a scene, even if those regions are zoomed, scaled[1], translated, or rotated in the viewing plane.

[1]Here, we distinguish between zoom or simple sampling resolution change, and "scale" as denoted by common literature in scale-space, meaning blurring

With those guidelines, and the experience described above, we are motivated to follow a philosophy using evidence accumulation, as the Hough transform does. Such a representation uses a great deal of memory, and performs most computations by lookup and simple arithmetic.

In this report, we deal only with two-dimensional silhouette images. Thus we only consider processes which are relatively low in the hierarchy of the human visual system. We make no claim that our algorithm "explains" human vision, only that it is highly parallel and therefore implementable in very fast hardware, and that it is not infeasible that something like this occurs in biological vision systems.

1.2 Evidence Accumulation by Transform

In this section, for pedagogical clarity, we begin with a simplest description of the algorithm and increase the complexity as issues such as nonuniform sampling occur. Furthermore, again for pedagogical clarity, we will describe each component in both discrete and continuous formulations.

The Hough and other similar methods form accumulator arrays whose peaks indicate properties of the scene. In this paper we initially (we will extend this in section 3) construct a two dimensional histogram of curvature and distance from the center of gravity (GC) of the region. We make use of this philosophy both to represent a region contour and to match shapes. We refer to the modeling and matching process as the "SKS algorithm."

1.2.1 Two-parameter, Discrete Formulation of the Model

Let $C = x_1, x_2, ..., x_n$ denote an ordered set of points circumscribing a region. For now, we assume no occlusion, so that $x_1 = x_{n+1}$. Then at any point on the contour in the viewing plane, $x_i = [x_i, y_i]$ we may determine its distance from the CG, $d_i = |x_i - x_0|$, and κ_i, the curvature of a small neighborhood[2] about point i, where x_0 is the vector-valued coordinates of the CG. The two parameters d and κ are chosen for their invariance to rotation and translation. We remind the reader that although curvature is theoretically invariant to rotation, the *estimation* of curvature is in fact sensitive to orientation. We will deal with this in section 3.

So for all n points on the boundary, we compute the curvature and the distance, and accumulate $(,\kappa)$ pairs, using, for each i,

$$A(d_i, \kappa_i) = A(d_i, \kappa_i) + 1 \tag{1}$$

After the accumulation, the array A is a model of the shape of the region. Experimentally, one discovers rapidly that this does not work well, as we will discuss in section 3, but the description is adequate for understanding the underlying philosophy of the approach. A bit more insight can be gleaned by restating the algorithm in continuous form.

[2]Further in the paper, we will shorten this description and refer to the "curvature AT point i"

1.2.2 Two-parameter, Continuous Formulation of the Model

The contents of the accumulator array can be written assuming both the contour and the parameter space are continuous. In this case, an integral formulation is required. This formulation also provides insight into the behavior of the algorithm. This formulation will be presented in detail and discussed in section 3 We initially represent the boundary of an object by a closed contour $C(s) = \begin{bmatrix} x(s) \\ y(s) \end{bmatrix}$, where s denotes arc length, and we will henceforth use the vector notation $\boldsymbol{x} = [\ x \quad y\ ]^T$to denote the position vector. For the time being, we will assume no points are occluded and the boundary is therefore closed. We will relax this assumption later. For each value s, one may compute the curvature $\kappa(s) : [0,1] \rightarrow \Re$ of the contour at that point. For any particular value of s, we may define a mapping of $\kappa(s)$, $\upsilon(\kappa(s)) = \left|\ \upsilon_x(\kappa(s)) \quad \upsilon_y(\kappa(s))\ \right|$ the magnitude of a vector from a point where the curvature has value κ to the center of gravity (CG) of the object. The composition $\upsilon(\kappa(s)) = [\upsilon_x, \upsilon_y]^T$, unfortunately, is not invertible, since there are likely to be multiple points on the boundary which have the same value of κ . κ thus actually indexes a density function[3] $p(\kappa, d)$, denoting the density of points with curvature κ which are a distance $d = |\upsilon|$ from the CG.

Since $p(\kappa, d)$ represents the number of times a point on the boundary has curvature κ and is a distance $|\upsilon|$ from the CG, we may compute this density function by integrating over the boundary of the object,

$$p(\kappa, d) = \frac{1}{P} \int_{s \in C} \delta(\kappa - \kappa(s), d - |\upsilon(s)|) ds \tag{2}$$

where again, $\kappa(s)$ denotes the curvature of the point s on the boundary, and δ is the usual Dirac delta function. This density function will be our model for the shape of a region.

1.2.3 Model Matching

Matching a model p_i to a contour C_j is possible by integrating the model over the contour. The result will be sharply peaked if the model is consistent with the contour. The matching process will be discussed in detail in section 3.4.

1.3 Terminology

Two terms are defined ambiguously in the literature: *scale* and *similarity*. The two definitions are *scale*, in reference to a region in an image, are

- The size of the region, computed by comparing the diameter of the observed region, in pixels, to that of some standard model of the same region. This is equivalent to *zoom* or *magnification* or

[3] Although this function has many of the properties of a probability (values between zero and one, integrates to one), we prefer to not call it a probability since there is no easily identified random process with which to associate it.

- The degree, σ, to which the region is blurred, assuming the observed region is the result of convolving some "ideal" region with a Gaussian with standard deviation σ. This definition is the one usually used in discussions of "scale space."

The confusion occurs because if one subsamples an image (effectively using larger pixels), the representation of a particular region is effectively blurred. In this paper, our use of the term *scale* will be equivalent to *magnification*.

The second term which we must carefully define is *similarity*, specifically in it use in the expression *similarity transform*. Suppose $C_1(s) : \Re \rightarrow \Re \times \Re$ is a curve in the plane, C_2 is another curve in the same space, and T is a transformation, $C_2 = T(C_1)$. Now suppose under the transformation, T, the length of all vectors is preserved. Then one would think that T should be called a *congruence transformation* by analogy to congruent triangles. If, however, C_2 is a magnified version of C_1, then T would be a *similarity transform*, again by analogy to similar triangles. Thus, translation and rotation (in the plane) are congruence transforms, and by extension, similarity transforms. However, zoom is a similarity transform, but not a congruence transform. Unfortunately, one will occasionally encounter usage of the term *similarity transform* in the literature where the author really meant *congruence transform.*

2 Background

The Hough Transform was originally published in a patent disclosure [16], and received considerable attention in the Machine Vision Community. In 1981, Deans[11] recognized that the Hough transform, applied to detecting straight lines, was equivalent to the Radon transform. That is, if $f(x, y) : \Re \times \Re \rightarrow \{0, 1\}$ is binary and equal to one iff the point x, y is on the contour, the integral

$$H(\rho, \theta) = \int \delta(f(x, y) - \rho(xcos\theta + ysin\theta))dxdy \tag{3}$$

produces a function, $H(\rho, \theta)$ of ρ and θ in which all colinear points in x, y map to peaks. The two-dimensional function H is referred to as the "Hough transform" or the "parametric transform" of the edge image $f(x, y)$.

Ballard [2] recognized that the concept of an accumulator could be used to recognize arbitrary shapes. First, a model of each object to be recognized is built: To construct such a model, at each point x_i on the boundary, measure the angle between the tangent and the vector to the CG, (see figure 1). Denote the vector as v_i and the angle as θ_i. Quantize θ into reasonable-sized bins. A table is then constructed with rows corresponding to the values of θ and on each row, a list of all observed values of the vector v_i. Matching an object to a model then consists of traversing the boundary of the contour of the object, and at each point:

- measure the angle between the normal at this point on the object contour and the CG of the object. Find the row in the model corresponding to this value.

- For each vector value on the list on this row, compute where the CG should be (coordinates of the boundary point plus the vector), and increment that point in an accumulator. The point with the maximum number of increments is the best match.

This algorithm, called the "Generalized Hough Transform" (GHT) is independent of rotation and translation. Invariance to translation is gained by simply referencing every point to the center of gravity of the region. Invariance to rotation is implicit in the definition of θ which measures a local property: angle between the local tangent and the vector from the local point to the CG. (We refer to this angle as the *local tangent orientation*). The GHT is not invariant to zoom or partial occlusion. In section 3 we provide an extended approach which is invariant to zoom and partial occlusion, and in section 8 we suggest a new result which is even more robust under zoom. One may choose to think of the countour as continuous, with piecewise-continuous ranges $\boldsymbol{v}(s)$, of the vector from point s to the CG, and $\theta(s)$, the angle between the tangent and the vector $\boldsymbol{v}$. Then, the GHT can also be written in continuous form as

$$H(v,\theta) = \int \delta((\boldsymbol{v} - ||\boldsymbol{x} - \bar{\boldsymbol{x}}||), (\theta - \theta(\boldsymbol{x}))d\boldsymbol{x} \tag{4}$$

where the vector $\boldsymbol{x}$ is the spatial coordinates of a point, $\theta(\boldsymbol{x})$ is the angle at that point, and $\boldsymbol{v}(\boldsymbol{x})$ is the vector from that point to the CG.

2.1 Computing Curvature

Given a curve in 3-space,

$$\mathbf{f}(s) = (\ x(s) \quad y(s) \quad z(s)\ )^T\ ,$$

parameterized by arc length, s, then the unit tangent vector $\tau = \frac{\partial \mathbf{f}}{\partial s}$. The tangent vector may also be differentiated with respect to s. The resulting vector $\boldsymbol{\tau}'$ may be normalized by dividing by its magnitude,

$$\mathbf{n} = \frac{\boldsymbol{\tau}'}{|\boldsymbol{\tau}'|}\ . \tag{5}$$

We refer to the magnitude of $\boldsymbol{\tau}'$as the *curvature*, $\kappa = |\boldsymbol{\tau}'|$ and the vector $\mathbf{n}$ as the normal vector.

The curvature is invariant to similarity[4] transforms, that is, any operation in the Lie group SE3. Not only is curvature invariant to similarity transforms, it scales inversely with zoom.

Curvature is a ideal property of any curve because of its invariance to rotation and translation. Curvature calculus and its properties are clearly defined in classical mathematics. However, the application of these properties on discrete data is not straightforward. This is because exact information about the continuous object is lost during the digitization process and therefore, curvature cannot be calculated accurately. This section contains an overview of the various discrete curvature estimation techniques found in literature. The state of the art of these methods is analyzed further in detail, and some experiments are described in section 5.1

[4]A similarity transform corresponds to a rigid body motion in 3-space, a motion which does not change the length of vectors. That is, translation and rotation, but not zoom.

2.1.1 Continuous Curvature

In two dimensions, the curvature of a curve at a point is defined as the directional change of the tangent at that point. There are various other definitions of curvature as defined in [33],[6].These are described below.

Consider a curve $\boldsymbol{x}(s)$ parameterized by arc length s.

Definition 1(Second Derivative Based Curvature)

As mentioned above,

$$k(s) = \begin{cases} +\left\|\boldsymbol{x}''(s)\right\| \text{ (Contour locally convex)} \\ -\left\|\boldsymbol{x}''(s)\right\| \text{ (Contour locally concave)} \end{cases} \tag{6}$$

where $\boldsymbol{x}''(s)$ is the second derivative of $\boldsymbol{x}(s)$.

Definition 2(Tangent Orientation Based Curvature)

$$k(s) = \theta'(s) \tag{7}$$

where $\theta(s)$ is the angle made by the tangent at $\boldsymbol{x}(s)$ with a given axis.

Definition 3(Osculating Circle Based Curvature)

$$k(s) = \frac{1}{r(s)} \tag{8}$$

where $r(s)$ is the radius of the osculating circle at $\boldsymbol{x}(s)$.

These definitions are equivalent in the continuous space. However, in the discrete space they lead to different algorithms.

2.1.2 Discrete Curvature Estimation

A complete analysis of the various discrete curvature estimation techniques is available in Worring and Smeulders [33], which concludes by recommending that the best method to find discrete curvature is by differential filtering of tangent angle using a Gaussian kernel.

$$k(i) = \frac{\theta(i) * G'_\sigma}{1.107} \tag{9}$$

$$\theta(i) = \tan^{-1}\left[\frac{y_{res}(i+1) - y_{res}(i)}{x_{res}(i+1) - x_{res}(i)}\right] \tag{10}$$

x_{res} and y_{res} are the resampled versions of the points on the curve (x, y). 1.107 is the average distance between two points on a discrete contour and is a multiplicative bias correction. The resampling is done on a straight line joining two adjacent points on the curve. Worring [33] states such a procedure will reduce the errors due to non-uniform sampling.

Vialard[32] suggests an improvement to the above method where the tangent is estimated in a purely discrete way using Digital Straight Segments. However, the biggest problem with these methods is the estimation of the Gaussian parameter σ. σ must be determined based on the nature of the data.

Coeurjolly[6] proposes a method where the curvature estimation is done using a purely discrete way using osculating circles which avoids any parameters based on the nature of the data. This method is explained in detail below.

2.1.3 Digital Straight Segments

A Digital Straight Segment(DSS) is defined using two support lines as follows:

$$D_{a,b,\mu,\omega} = \{(i, j) \in Z^2 : \mu \leq ai + bj < \mu + \omega\} \tag{11}$$

where a/b is the slope. a and b are relatively prime integers. μ is the approximate intercept and ω is the arithmetic width.

If $\omega = max(|a|, |b|)$, then the lines are called *naive* lines. These lines are 8-connected lines in Z^2. If $\omega = |a| + |b|$, then the lines are called *standard* lines. These lines are 4-connected lines. The term *connected* here implies adjacency. The connectivity is not absolutely necessary. A set of points are part of a straight segment as along as they lie within the support lines. Therefore, if two points are not connected and lie within the support lines then it is possible to find 4 or 8 connected points joining them.

It is interesting to note that *standard* lines are a bit more resistant to noisy points than *naive* lines because of greater separation between the support lines. *Standard* lines are used for curvature estimation in our algorithm.

There are two important algorithms found in the literature to find Digital Straight Segments. For *standard* lines, the algorithm is defined in [20]. Recognition of *naive* lines is discussed in [12]. In

section 3.1 we describe the algorithm to find *standard* lines as described in [20].

2.2 Matching

Shape matching is a well-researched field documented in a plethora of publications. The author's text [31] provides tutorial presentation of many of these, up to 2002, and an excellent survey by Zhang and Lu [34] brings the reader up to 2004.

The algorithm described in this paper characterizes each boundary point by the distance from the geometric center, the curvature at that point and (optionally) by the angle between the vector from the geometric center and the tangent at that point. Using the Digital Straight Segments algorithm (discussed in section 3) to compute the curvature provides the tangents as a component of the calculation Therefore the third feature is obtained at minimal additional computation.

2.3 Wetware

Research [24][23][25] in the visual neurosciences shows this approach to shape representation is adopted by primate cortices (in the higher visual areas, V4-IT). Connor in [8] discusses a parts based representation of boundary fragments and documents that response of neurons in the visual cortex of a macaque. These neurons in the V4 show sensitivity to the curvature, the radial position and the context (connectivity to other parts).

We will base one version of our algorithm on Principal Comonents Analysis, and the reader is referred to [22] to address the issue of biologically-plausabe mechanisms for performing this computation.

3 Approach

In the current work, we have extended the philosophy of the GHT by using different parameters, leading to more robust representations and invariance to partial occlusion.

Several variations of the algorithm were evaluated over the investigation; these will be mentioned after the descrription of the parameters.

First, we discuss use of Digital Straight Segments to find curvature.

3.1 Algorithm for Recognition of Standard Lines

The algorithm was initially proposed in [20]. The description given below is based on [19]. The algorithm is based on the principle that a set of points (x_i, y_i) are part of a DSS if:

$$0 \leq bx - ay + c \leq |a| + |b| - 1 \tag{12}$$

where a, b and c are integers with a and b being relatively prime. The algorithm assumes that the points are 4-connected.

1. $p_N = q_P = (x_1, y_1), q_N = p_P = (x_2, y_2), a = x_2 - x_1, b = y_2 - y_1$
2. $n = 3$
3. Repeat steps below until (x_n, y_n) are not DSS.
4. $c = b * x_{n-1} - a * y_{n-1}$
5. $r_n = (x_n, y_n)$
6. $h = b * x_n - a * y_n - c$
7. if $0 \leq h \leq |a| + |b| - 1 : (x_n, y_n)$ is a part of the DSS.
8. if $h = -1$ then $q_N = r_n, p_P = q_P, (a, b) = r_n - p_N$. (x_n, y_n) are part of the DSS.
9. if $h = |a| + |b|$ then $q_P = r_n, p_N = q_N, (a, b) = r_n - p_P$. (x_n, y_n) are part of the DSS.
10. Otherwise, (x_n, y_n) are not DSS. Stop at previous vertex (x_{n-1}, y_{n-1}).

3.2 Estimation of Curvature and Tangents Using DSS

Digital Straight Segments can be used to estimate discrete curvature. One of the easiest and simplest way to do this is presented by Coeujolly and Svensson [7]. Let $v_1 = (x_1, y_1)$ and $v_2 = (x_2, y_2)$ be the end points of the maximum length DSS around point v_0 on the curve. Note that v_1 and v_2 are also points on the curve. Let a be the Euclidean distance between v_0 and v_1. Similarly, let b be the distance between v_0 and v_2 and c be the distance between v_1 and v_2.

Then, the radius of the osculating circle at v_0 can be approximated by finding the radius of circumcircle(R_c) of the triangle with vertices v_0, v_1 and v_2.

$$R_c = \frac{abc}{4A} \tag{13}$$

$$A = \frac{\sqrt{(b+c)^2 - a^2}.\sqrt{a^2 - (b-c)^2}}{4} \tag{14}$$

A is the area of the triangle. Curvature at v_0 is, therefore, approximated by:

$$k = \frac{1}{R_c} \tag{15}$$

The line joining v_1 and v_2 is the approximation of the tangent at v_0.

We follow this approach in our curvature estimation work. Experimental results in curvature representation and estimation are discussed in Section 5.1.

3.3 Construction of the SKS model

In this section, we describe how the invariant shape model is developed. Two versions are discussed. The first utilizes two parameters, distance from the CG and curvature. We have also experimented with orientation of the principal axis, which allow the introduction of a third parameter, orientation.

Denote by $\kappa(s)$ the curvature at the point where the arc length equals s. At the same point, define the vector to the center of gravity $\boldsymbol{v(s)}$, the orientation angle θ, and the tangent angle ψ. We observe that both the length $d = |\boldsymbol{v}|$ and κ are invariant to translation and rotation, θ is invariant to translation and zoom, and ψ is invariant to all similarity transforms. These angles and lengths are illustrated in figure 1.

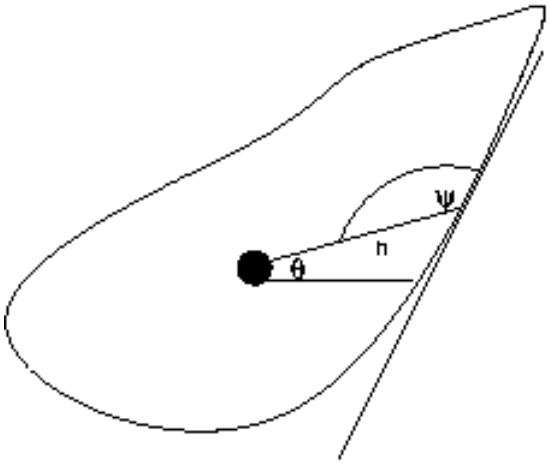

Figure 1: The feature space, Distance d, curvature κ and tangent angle ψ, and orientation angle θ.

In the two parameter, discrete representation, we consider the curve parameterized by $l \in Z$, then the model of curve i depends on κ and d,

$$p_i(\kappa, d) = \frac{1}{N} \sum_k \delta(\kappa - \kappa_l, |d_l - d|) \ , \tag{16}$$

where $d_k = |x_k - x_{CG}|$ is the distance from the k^{th} on the boundary to the CG, with N points on the contour.

In the continuous representation of the same two-dimensional form, we think of an integral over a contour C which has perimeter P, and p in this case is

$$p_i(\kappa, \psi) = \frac{1}{P} \int_{s \in C} \delta(|\kappa - \kappa(s)| \, , |\psi - \psi(s)|) ds \tag{17}$$

The three-parameter version of the model is generated by populating the feature space by estimating the (κ, d, θ) tuple at each point. The three features are scaled and quantized. The bin represented by the quantized values is then incremented, just as in the two-parameter case. This is repeated for all the points in the shape. Owing to the fact that the model determined by each and every point, we normalize the accumulated model by the number of points to obtain a percentage representation. The model can now be viewed as a density function. It represents the fraction of points that satisfy a particular combination of features. The density function corresponding to the i^{th} shape can then be represented as:

$$p_i(\kappa, d, \theta) = \frac{1}{P} \int_C \delta(|\kappa - \kappa(s)|, |d - d(s)|, |\theta - \theta(s)|) ds \tag{18}$$

where as before, the contour has a perimeter of P. Here, $\delta(x, y, z)$ is the Dirac (delta) function and the integral is over the contour $C(s)$.

Computationally, any representation must ultimately be cast in a discrete form, as a discrete contour $C = \{C_1, C_2, \ldots, C_l, \ldots, C_N\}$,

$$p_i(\kappa, d, \theta) = \frac{1}{N} \sum_l \delta(|\kappa - \kappa_l|, |d - d_l|, |\theta - \theta_l| \ , \tag{19}$$

where now, $l = 1, 2, \ldots N$ is the (unit-spaced) distance from the starting point, and the Kronecker delta is required. Some of the shape information maybe lost and the representation may not be truly invertible. Quantization is a parameter in the algorithm and the resolution of the model space can be set to make the representation as unique as possible. The resolution of the model space determines the ability of the algorithm to generalize (and hence discriminate between similar shapes). At small resolutions the generalization will be good but this comes at the cost of discrimination. On the other hand at high resolutions the generalization gets progressively poor but discrimination improves drastically. Higher resolutions also imply larger computational complexity (both space and time). The resolution (100 x 100 x 180) was determined experimentally.

3.3.1 Model Smoothing

The delta function in Equations 17 and 18 produces a singularity which, in the discrete implementation, means that only a very few points, on a narrow locus, are incremented. This is most easily seen in figure 2 It is also instructive to observe two versions of the same contour to notice similarities. For example figure 3 is the same tank as in figure 2 on the left. To allow for a smoother representation, with more generalization ability, we have two choices: we could produce an averaged representation or a blurred representation.

To produce an averaged representation, we constructed a model for the same tank at fifteen degree rotations, and averaged the result, taking advantage of sampling noise to provide smoothing. This result is shown in figure 4. Superior results were obtained, however, by simply taking one model (which is rotation invariant up to the sampling noise anyway, and applying some sort of Gaussian smoothing filter. That is, the model could be convolved with a Gaussian kernel such as (in the two-parameter case):

$$\hat{p} = p * \frac{1}{\sigma} - e^{-\frac{(\kappa-\kappa_0)^2 + (d-d_0)^2}{\sigma^2}} \ . \tag{20}$$

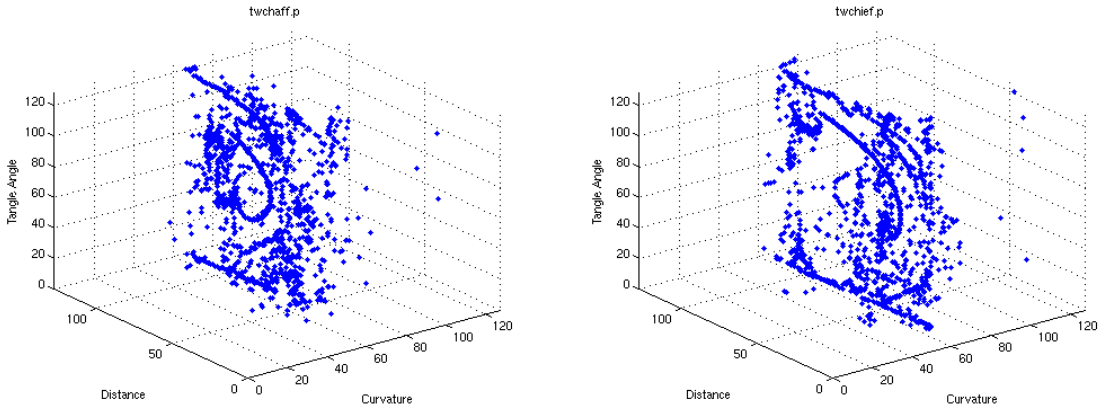

Figure 2: Models of two tanks. No averaging has been done

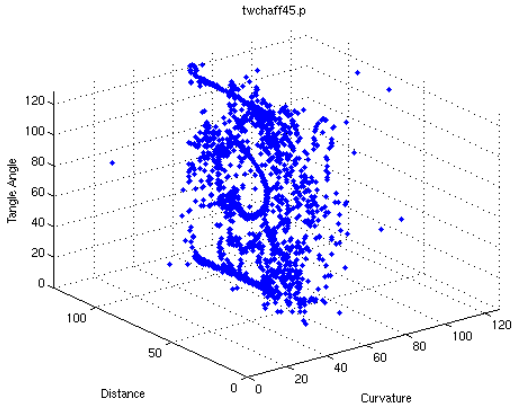

Figure 3: The nonsmoothed model of the same tank as the model on the left of figure 2 but rotated by 45 degrees.

However, a simpler strategy is to simply redefine the delta function to be a Gaussian:

$$\delta_\sigma(a,b) = \frac{1}{\sigma} e^{-\frac{a^2+b^2}{\sigma^2}} \tag{21}$$

where we observe that for reasonable values of a and b,

$$\lim_{\sigma \to 0} \delta_\sigma(a,b) = \delta(a,b) \; .$$

3.4 Matching

In this section, we describe how a model produced as a two- or three-dimensional histogram may be used to match to a shape in an invariant way. We discuss the two- and three-parameter versions of the algorithm together in this section.

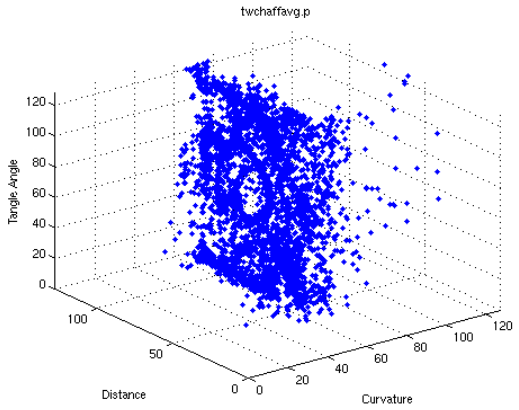

Figure 4: The model constructed by averaging the models of a number of models, each resulting from a pure roation. This should be compared to the left-hand model in figure 2.

3.4.1 Feature Quantization

In describing the model of a shape, we have used the term "density function," but of course, it is really an accumulator array – a two- or three-dimensional histogram. The 3-Dimensional feature space is digitized into 100 bins along the curvature axis, 100 bins along the distance axis and 100 bins along the tangent angle axis. In addition, to make the representation invariant to size of the object, the object is scaled to a uniform size prior to be binning process. In our experiments we scaled the object such that the farthest point in the object is at 128 units from the geometric center. This particular scaling strategy does not survive partial occlusion of that most distant point however.

The curvature at a point is also sensitive to the scaling. Hence, along with the distance from the geometric center, the curvature at each point is also scaled with respect to the maximum distance of the object (In the human visual system visual data is presented at different scales and research suggests a maximization function is computed to select the appropriate scale).

The scaling of the distance and curvature ensures that the model is independent of the size of the object. However, it also makes the model generation and recognition processes sensitive to occlusion of the maximally distant points.

The curvatures are scaled so that they all lie between 1.0 and -1.0. All singularities are squashed to either extremes as suggested by [24].

The number of bins for quantization were determined after a set of experiments where models were generated at different resolutions of the feature space. The 100 x 100 x 100 space was selected when it showed that at this resolution there was a reasonable balance between discriminability and generalization.

h

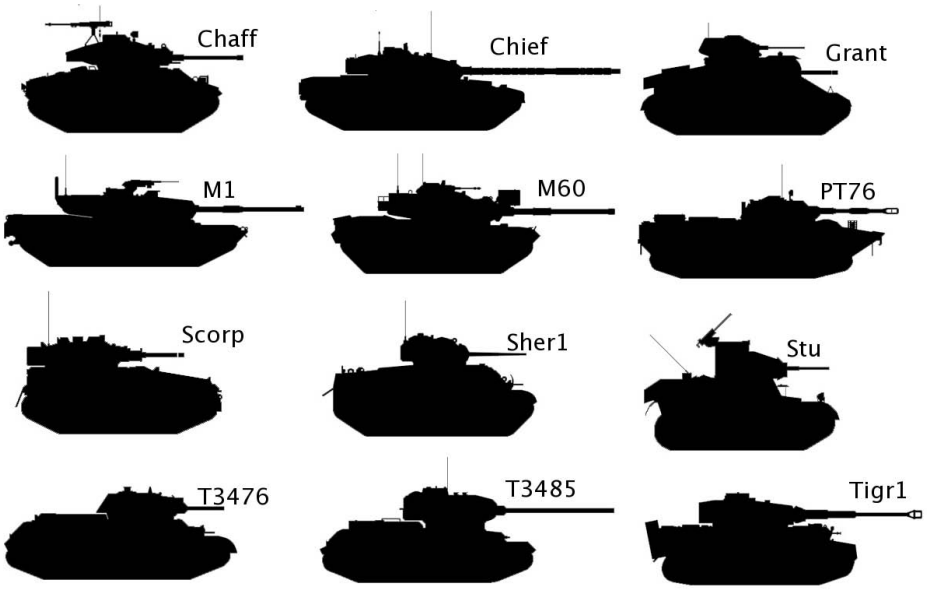

Figure 5: The twelve tanks, zoomed so that all silhouettes have the same area.

4 Data Sets

We used two data sets in this work: The first is that of a 12 tanks. We have also trained and tested the algorithm on a data set of 1100 fish contours.

4.1 Example Models of Tanks

The tank contours are shown in figures 5 The tank images are particularly interesting for this study because (surprisingly) they cause considerable problems for the algorithm compared to the fish images. This is primarily due to the fact that a great deal of the SKS algorithm is dependent on accurate computation of curvature. Since the tank images have so many straight lines, most of the points on the contour actually have a curvature of zero, or a curvature which cannot be accurately computed and is simply marked to "high." Nontheless, as we will discuss in the results section, the algorithm performs quite well.

4.2 The SQUID data base

We also tested the algorithm on the SQUID[5] data base. This data base has hundreds of contours. We used the thirty-one which are illustrated in Figure 6.

[5]http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html, used with permission.

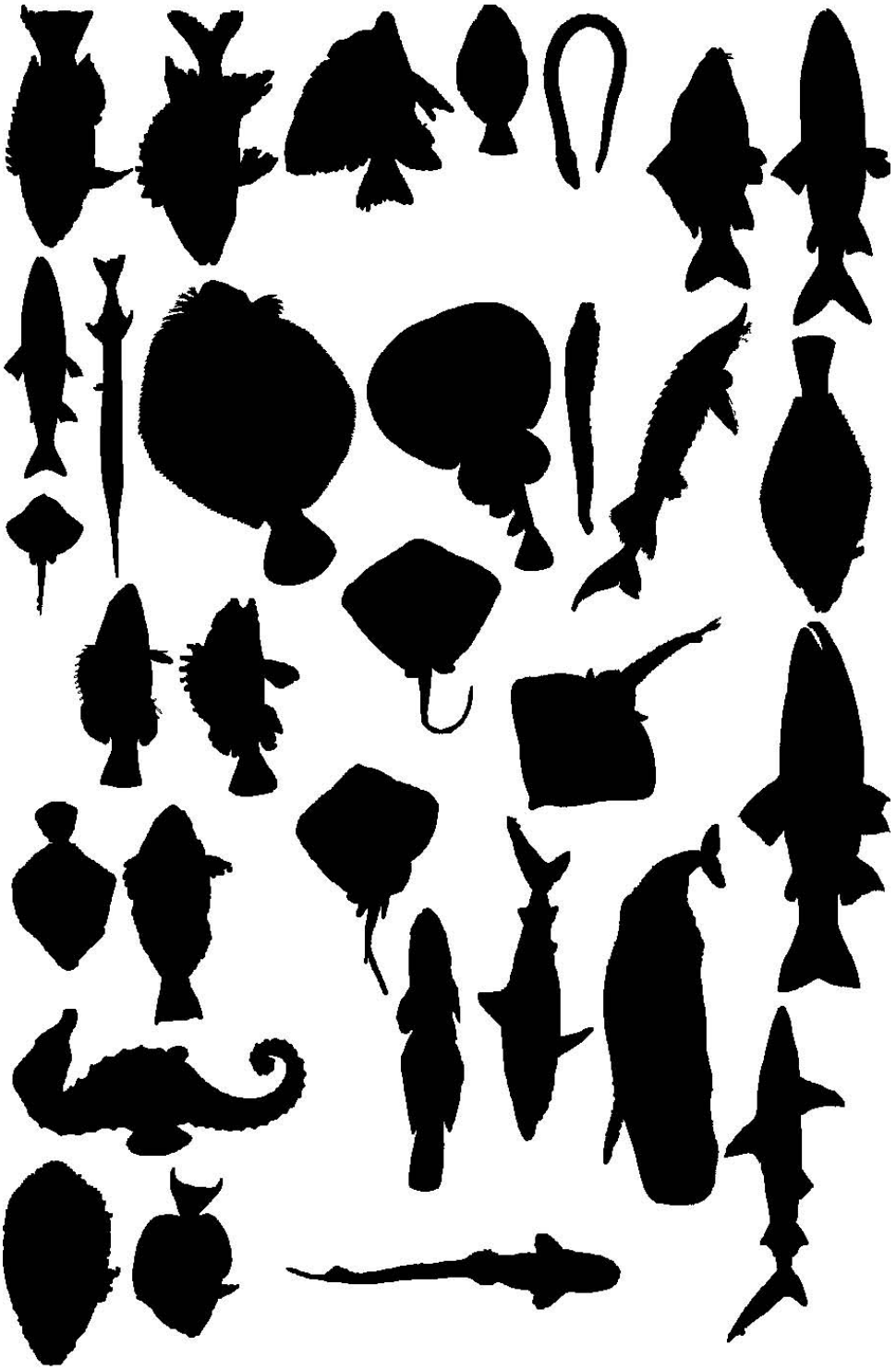

Figure 6: Thirty-one fish silhouettes from the SQUID data base, used in this project

5 Experimental Results

In this section we provide a collection of examples of phenomena which would be expected to degrade the performance of the algorithm, and show the algorithm performs well against them. In section 5.9, we will compare performance with other algorithms, including use of the Haussdorf distance and invariant moments.

5.1 Curvature

In this section, we test the accuracy of the DSS curvature estimator. Eleven circles of radii 5, 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 pixels were taken and their curvature was estimated using the algorithm. A plot of the average estimation error versus radius is shown in 7. The plot clearly

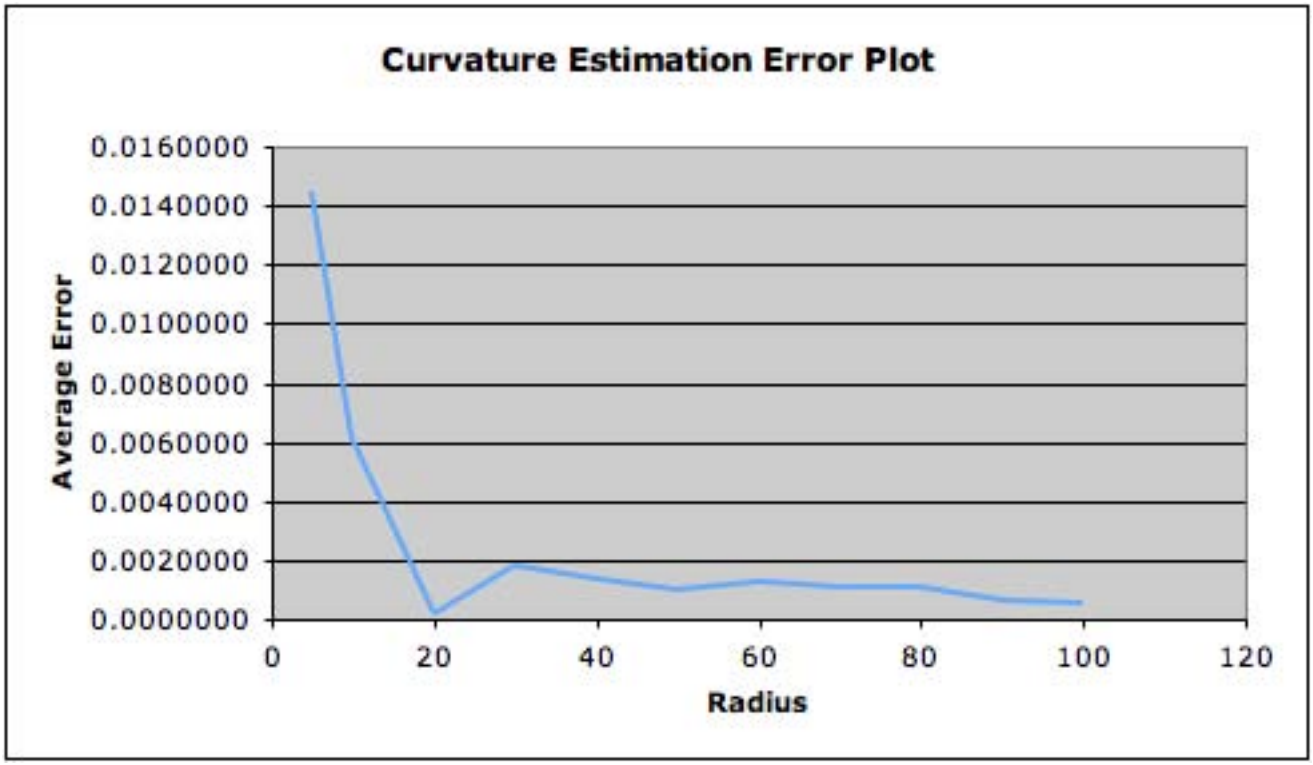

Figure 7: Average Curvature Estimation Error

shows the convergence of the average curvature to the true continuous curvature with increase in resolution. However, the convergence is true only for average curvature. The curvature at a point does not have asymptotic convergence. According to [10], the asymptotic convergence of a DSS curvature estimator is still an open problem.

However, the advantages of DSS curvature estimators clearly outweigh this disadvantage. The biggest advantage of DSS based curvature estimators is that they are independent of the nature of the data when compared to classical techniques. The algorithm is highly parallel and requires no parameters to set, giving very good accuracy at the same time.

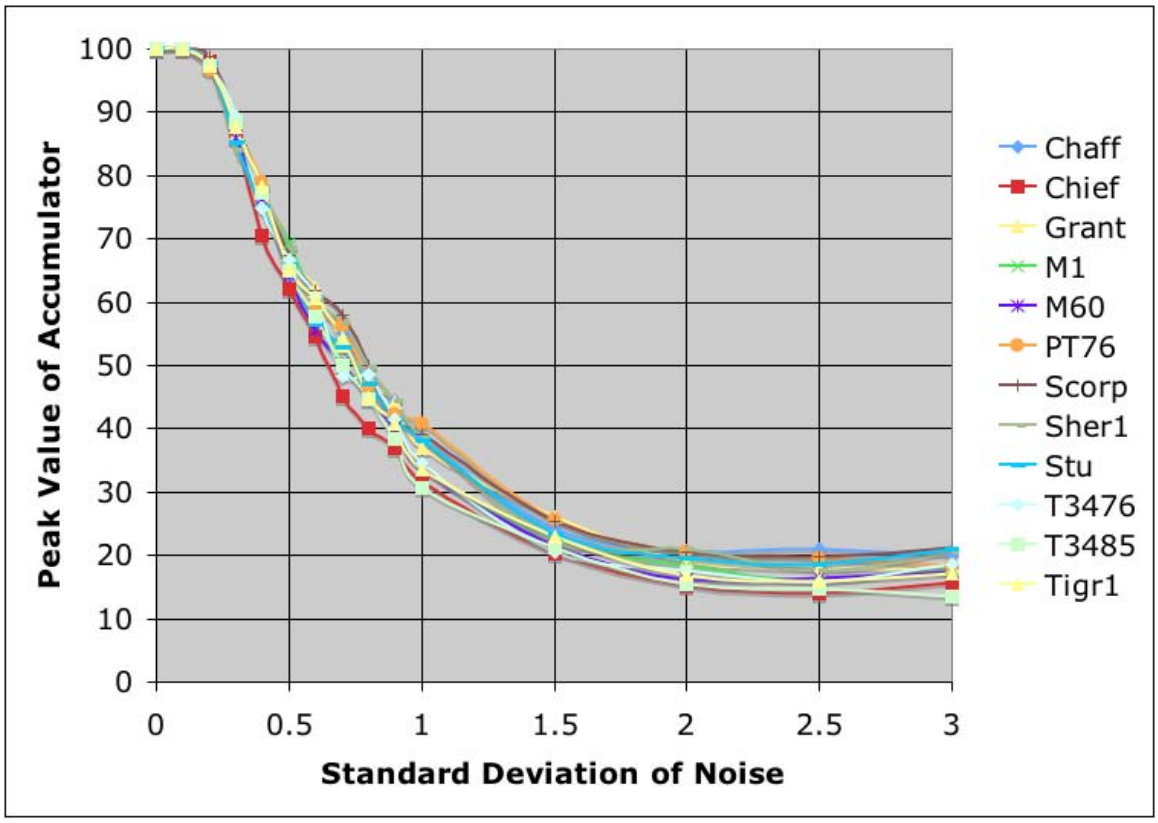

Figure 8: The match quality (normalized to 100.0) is plotted on the vertical axis. On the horizontal axis is the standard deviation of the Gaussian noise which was added to each spatial coordinate to cause jitter.

5.2 Degration Under Jitter

In this experiment, all tank images were compared to their own models, so performance should be excellent, except that a random (Gaussian) jitter was induced by adding random numbers to the x and y coordinates of each point in the image. The quantity measured is the peak intensity of the accumulator, normalized to 100.0. This is illlustrated in figure 8.

5.3 Effect of Smoothing the Model

In figure 9, we illustrate the importance of smoothing the model. As discussed in section 3.3.1, if we leave the model as developed by using a delta function, we obtain a model which is very crisp, and can only be matched by a contour which precisely (under the allowable transformations), matches the model. By replacing the model with a smoothed version of the itself, we get better generalization and better tolerance to noise. In the figure, the horizontal axis represents the standard deviation of Gaussian noise added to the coordinates of each point, simulating jitter. The individual curves represent the standard deviation of the Gaussian used to replace the delta function. All curves represent the result of matching TANKChaff with a distorted version of itself, and only reflect the

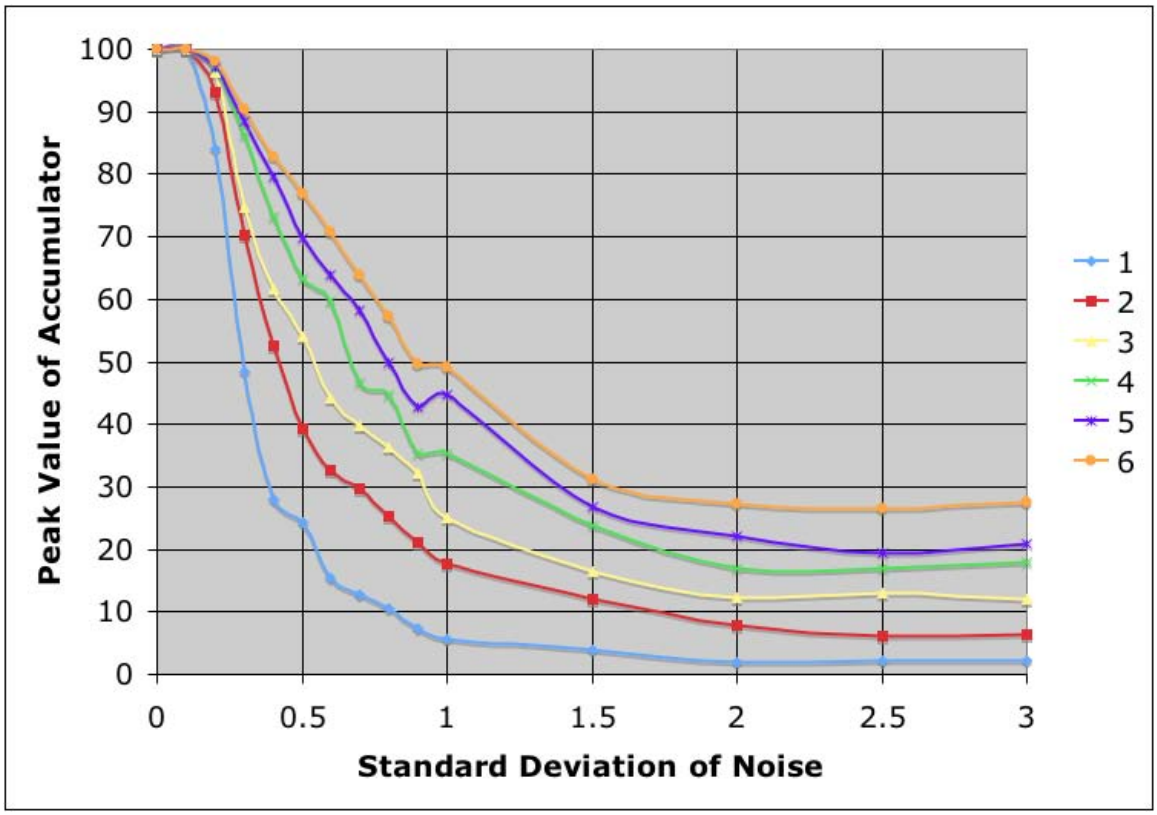

Figure 9: Results of matching one particular tank silhouette (TANKChaff) with itself, where the model has been blurred by a Gaussian of variance σ^2.

amplitude of the match quality – higher is better.

5.4 Sensitivity to Rotation Invariance to Model Smoothing

To study the nature of model smoothing on rotational invariance, matching of a model to rotated versions of itself were performed, under varying degrees of model smoothing. The results in figure 10 are for one tank (TANKChaff). Other tanks follow similar trends.

In figure 11, we illustrate a contour which has been synthetically jittered by adding a Gaussian-distributed random number with a standard deviation of 3.0 to each spatial coordinate. In order to ensure a closed contour results, it is then necessary to perform some sort of interpolation between the jittered points. This interpolation results in local curvatures which are more an artifact of the interpolation algorithm than the shape of the curve. Furthermore, there is no physical process that we can think of that would produce such distortions to an image. For these reasons, we have elected to not use jitter as a distortion in further work.

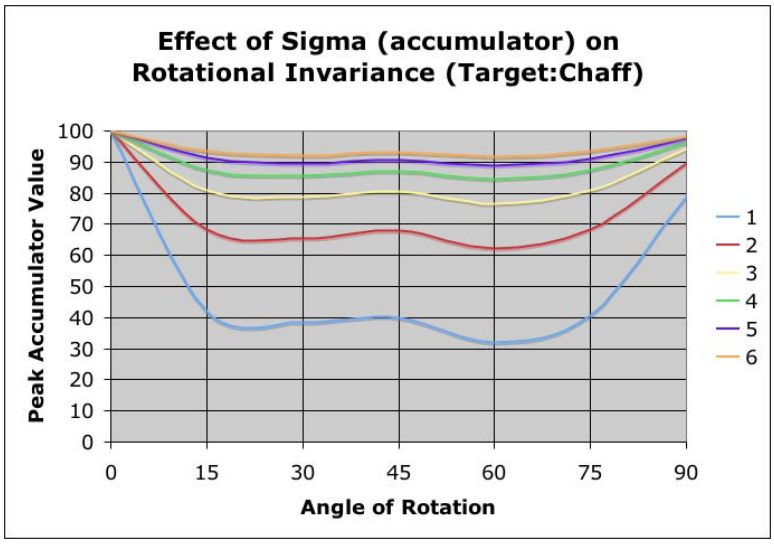

Figure 10: Results of matching one particular tank silhouette (TANKChaff) with rotated versions of itself. The different curves are for different degrees of model smoothing. Each curve indicates the model was smoothed with a a Gaussian of variance σ^2, where the color/σ relationship is given on the right

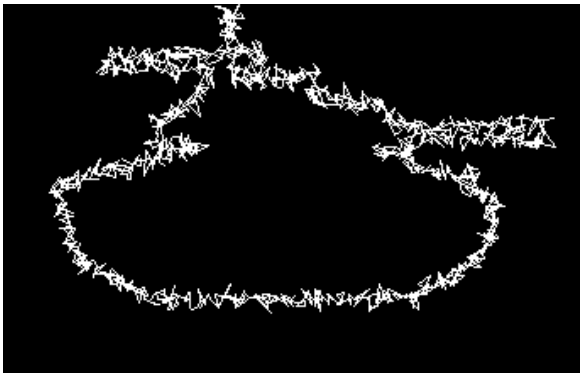

Figure 11: Distortion of the contour of TANKChaff by adding a Gaussian-distributed random number with standard deviation of 3 to each spatial coordinate.

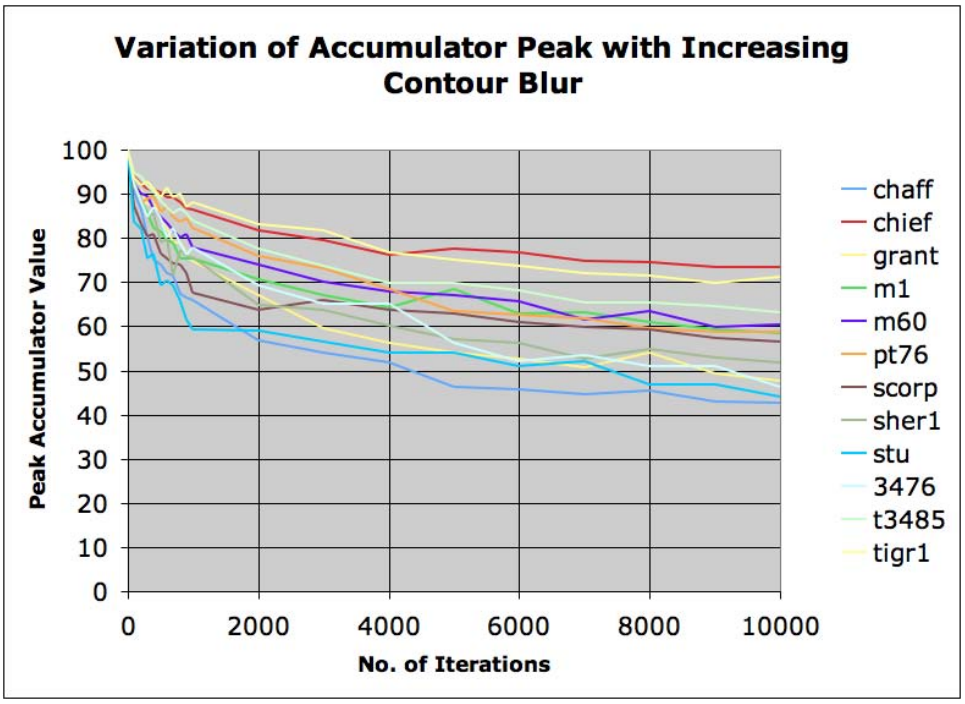

Figure 12: Degradation of peak amplitude with increasing contour blur. This particular graph was made using different degrees of contour smoothing with TANKChaff, however, similar results were found for both tanks and fish. See figure 20 for examples of contour smoothing.

5.5 Contour Blur

An artificial distortion much more realistic than jitter is *contour blur*. This distortion models the effect of low-pass filtering when the only input is the contour. This is accomplished by using the parametric form for the contour, $[x(s), y(s)]$, and smoothing (using digital convolution) each spatial coordinate using the same one-dimensional smoothing kernel:

$$[\hat{x}(s), \hat{y}(s)] = [x(s) \otimes h, y(s) \otimes h] \ . \tag{22}$$

One may accomplish blur in two ways: by using a large convolution kernel, say a Gaussian with a variance of 10.0, or by using a small kernel like a Gaussian with a kernel of 1.0, and repeating that smoothing operation numerous times. The second of these operations is reminiscient of a diffusion process which occurs over time. In fact, since the Gaussian is the Green's function of the diffusion equation, this is exactly correct. using the small kernel models diffusion by a short period of time, and repeating it is equivalent to simply using a large variance kernel.

Figure 12 illustrates the degradation in performance in matching a contour (TANKChaff) with blurred versions of itself. All tank contours showed similar behaviors.

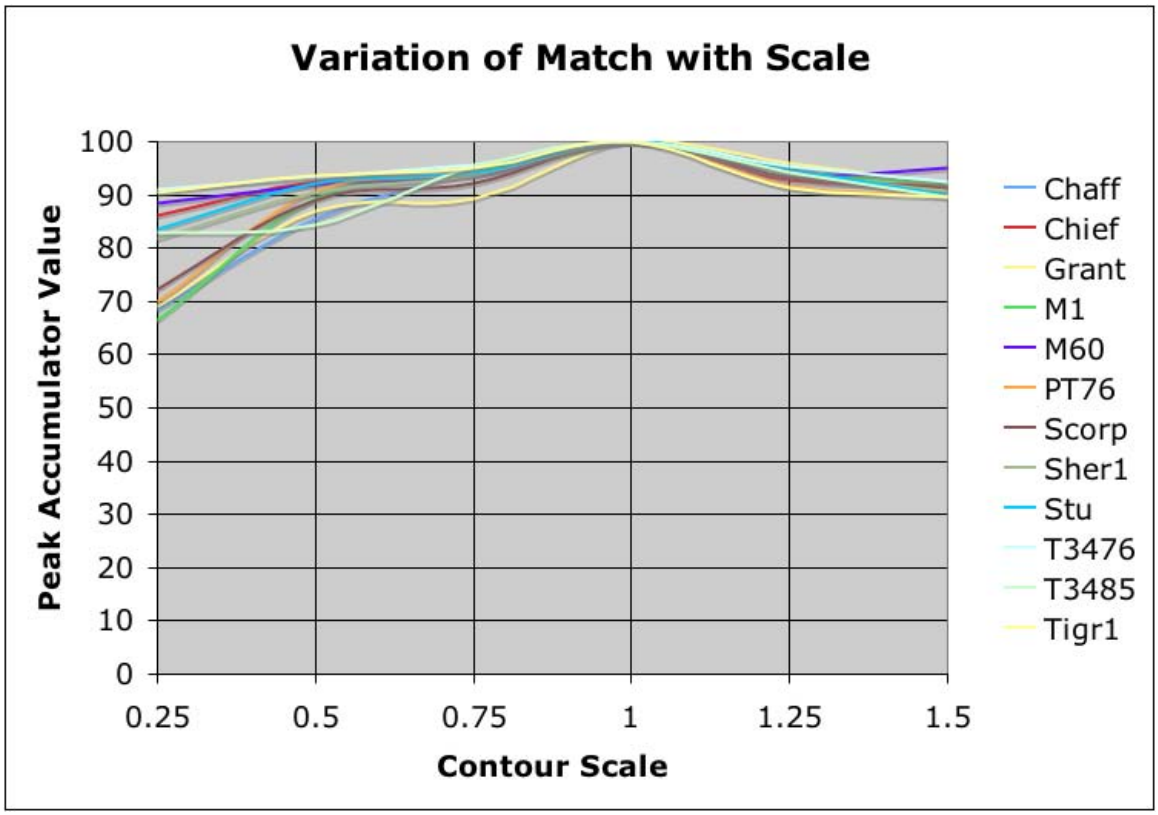

Figure 13: Each tank contour, after zooming down to 25% of original size or up to 150% of original size is compared with the model of itself.

5.6 Scale

In Figure 13 we illustrate the effect of scale on representation accuracy. In this experiment, each of the tank models was zoomed, down to a minimum of 25% of original size (in each dimension), and up to 150% of original size. The zoomed version of the contour is then matched against the model without change.

5.7 Partial Occlusion

In figure 14, we display and example of a partically occluded tank – matched against itself in the left image and against a different tank in the right image. Clearly the image on the left, the good match, produces a higher and sharper peak.

5.8 Similarity Detection

One of the desirable properties of the algorithm is its ability to detect similarities between shapes. Since the algorithm involves encoding the structural information (curvature) of a shape relative

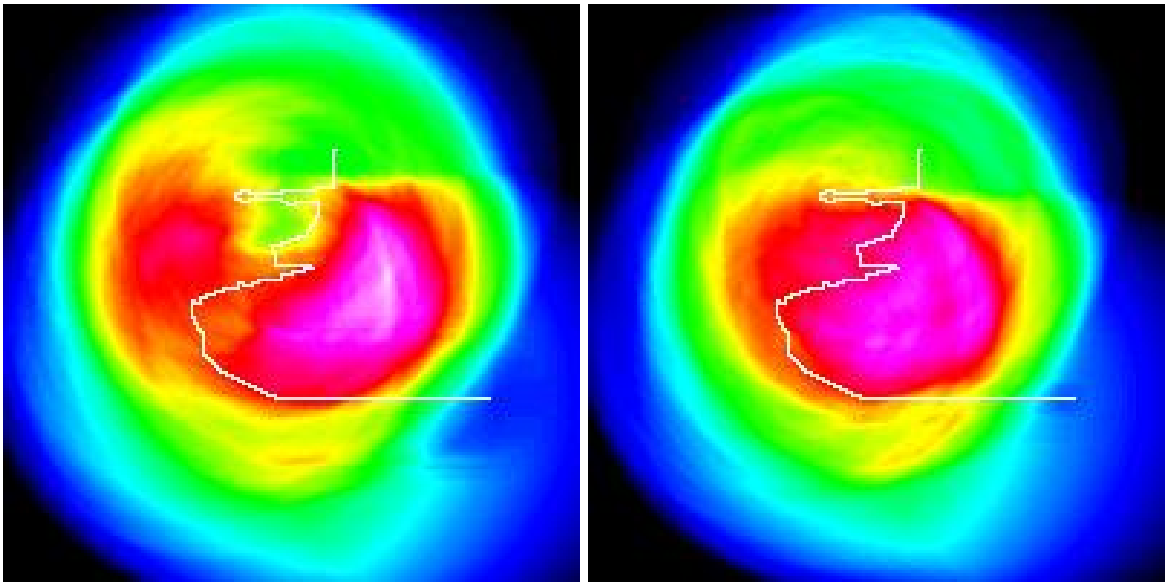

Figure 14: In the left figure, a contour (TANKChaff) with approximately 70% occlusion is matched against an unoccluded model of the same contour. The sharp peak is clearly seen. In the figure on the right, the same partially occluded contour is matched against a model of a different tank (TANK2). It is clear that the accumulator on the right does not possess the sharp peak which identifies a good match

to a set of global features (principal axis and geometric center), it is expected to produce similar models for shapes with similar structure. In the previous sections the ability of the algorithm to discriminate between somewhat similar shapes (all tanks) has been documented. In this section the ability of the algorithm to detect similar shapes is investigated. A formal evaluation of the output of the algorithm as a shape metric is still under consideration. However, preliminary evaluation of the algorithm in this regard is presented here.

The SQUID data set was sampled and 31 contours were selected randomly. Models were generated for each of the contours. Next, the entire data set of 1100 contours was matched against the models. The top five matches for each model were isolated. An artificial threshold of t was set and the number of contours with peak accumulator value above t for each model was calculated. Some of the matches are shown in figures 15 - 16. Table 1 shows the number of contours (of the total 1100) with matches above the threshold t for different values of t.

Table 1 shows the number of contours with accumulator peaks higher than a particular threshold (t) when matched against each of the 31 models. Some of the models are more "similar" to other shapes in the data base, as can be clearly seen from the table. For example contour kk492 over 100 contours with an accumulator peak of greater than 60. The same is not true for other contours. The contour kk127 had all peaks below 50 (excluding itself). In order to attain a qualitative feel for how similar some of the matches are the the top matches for some of the models are shown in figures 15-16. In all the figures the shapes that show a high degree of match (in terms of accumulator peaks) are visually very similar to the model. For example, all the shapes on the right of figure 16 are sea-horses. Similarly, all the shapes on the right of figure 15 look like sting rays, demonstrating the ability of the algorithm to detect similarity between shapes.

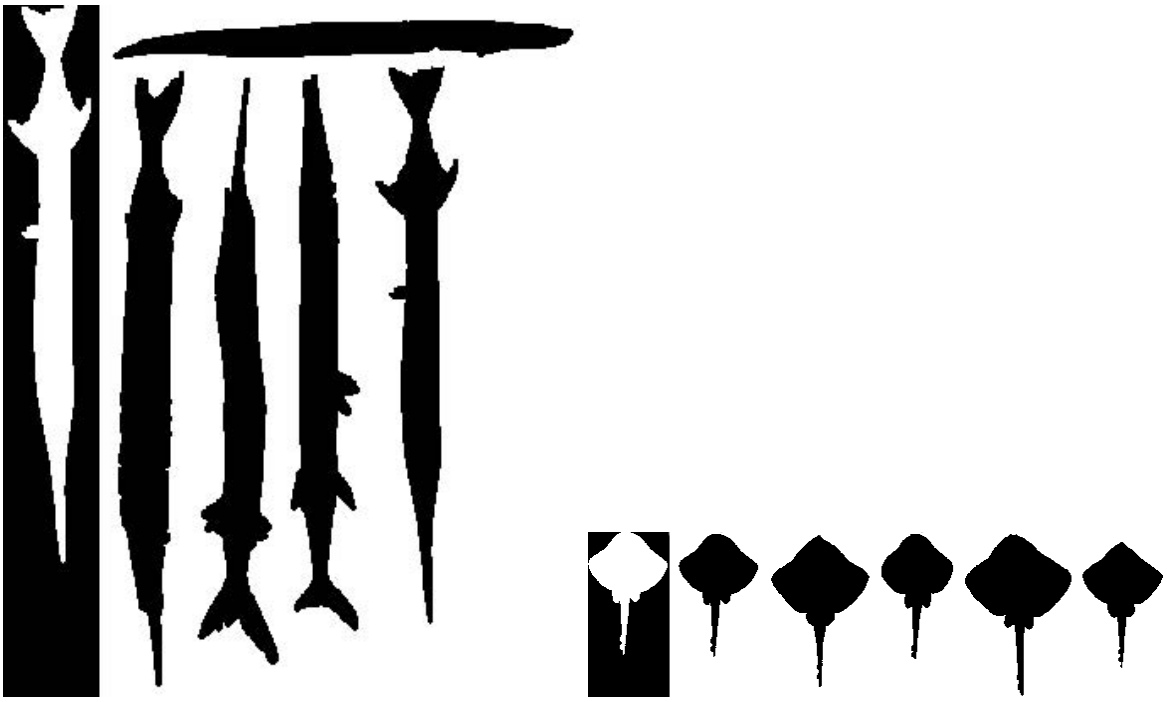

Figure 15: Best five matches (black contours) from the SQUID data base for the model (white on black background) of contours kk87 and kk732.

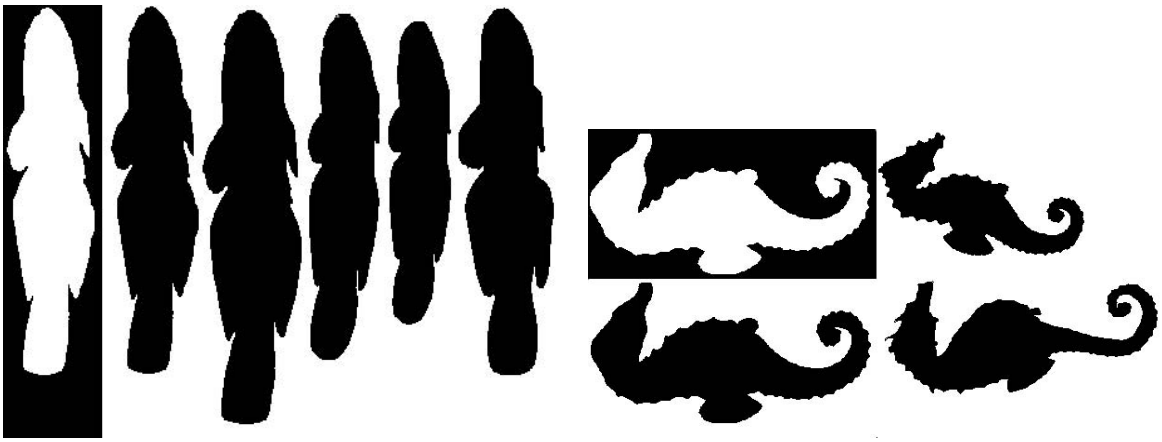

Figure 16: (Left) Best five matches (black contours) from the SQUID data base for the model (white on black background) of contour kk942. (Right)Best three matches for contour kk779. All other matches were below the threshold of 60

Model	t = 90	t = 80	t = 70	t = 60	t = 50
kk5	1	1	5	66	289
kk19	1	1	3	28	135
kk30	1	1	2	31	80
kk42	1	1	1	12	22
kk72	1	1	9	48	183
kk83	1	1	5	78	384
kk87	1	5	16	31	48
kk100	1	1	1	12	162
kk127	1	1	1	1	1
kk184	1	1	1	1	1
kk209	1	1	1	3	58
kk250	1	1	1	4	17
kk263	1	1	3	7	34
kk273	1	2	9	38	105
kk310	1	1	1	12	86
kk378	1	1	2	5	19
kk430	1	3	8	36	136
kk464	1	1	1	3	14
kk473	1	1	1	1	5
kk480	1	1	1	7	183
kk495	1	2	8	138	445
kk502	1	3	25	122	359
kk558	1	3	6	50	231
kk676	1	1	1	1	5
kk689	1	1	7	50	210
kk717	1	1	3	22	169
kk732	1	8	11	16	29
kk766	1	4	31	146	370
kk779	1	1	2	3	19
kk810	1	1	5	33	234
kk942	3	5	33	136	366

Table 1: The number of matches above the threshold (columns) for different models (rows) when the 1100 contours of the SQUID data base were matched against the models.

5.9 Comparison with Other Shape Matching Methods

To date, we have only compared the performance of the SKS algorithm with use of the Haussdorf distance [1][3] and with the Hu[21] invariant moments. Since the Haussdorf distance is not invariant to congruence transforms, and the SKS algorithm is, this is not a fair comparison – our algorithm is much more general. However, if we restrict the comparison to only untransformed images, we can still evaluate the performance to some degree. Figure 17 illustrates how the Haussdorf distance varies with Gaussian jitter. Observe that the Haussdorf distance is relatively immune to Gaussian

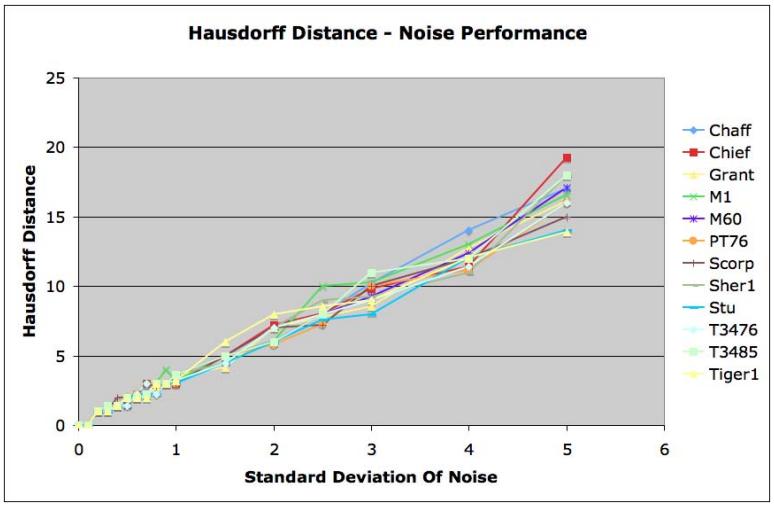

Figure 17: Haussdorf distance between Tank 1 and a distorted version of itself, as a function of jitter.

jitter, for small values of jitter, and is linear with the jitter. This is probably an artifact of the way we jitter the image, by randomly moving the points, but then interpolating between them.

5.9.1 Matching as a Function of Contour Smoothing

The thirty one shapes of figure 6 were used to construct the 31 corresponding models. Then, the 31 contours were smoothed using Equation 22, and matched against each of the models. The correct classification rate is shown in Figure 18. We observe that random guessing would produce a correct classification rate of 3%. To get a feel for the distortion introduced by contour blur, we provide the contour of the first fish image (Figure 20) after 5000 and 10000 iterations of smoothing. Figure **??** provides a comparison of the SKS algorithm and the Hu invariant moments as a function of contour blur. The performance of the Haussdorff distance is also included in the graph, but we remind the reader that the Haussdorf distance is not invariant to zoom, or rotation. We observe that the SKS algorithm is sensitive to contour blur, but provides significantly higher accuracy rates, often twice as good.

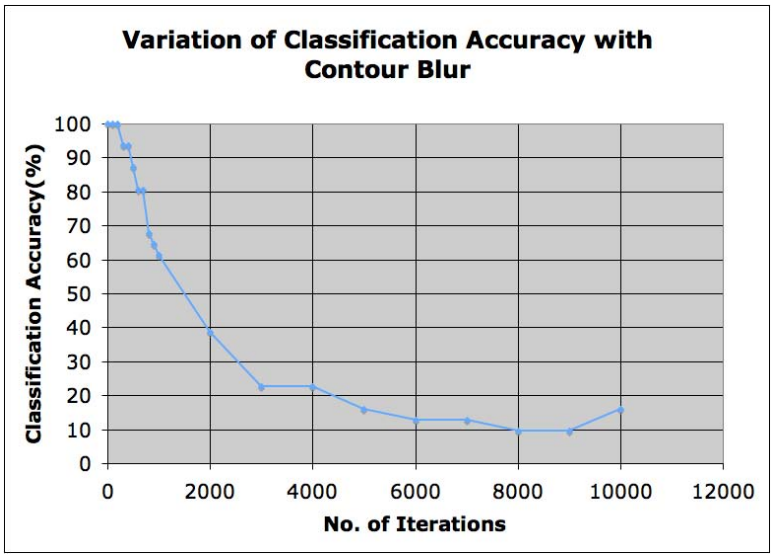

Figure 18: Percentage of correct classifications as a function of contour blur

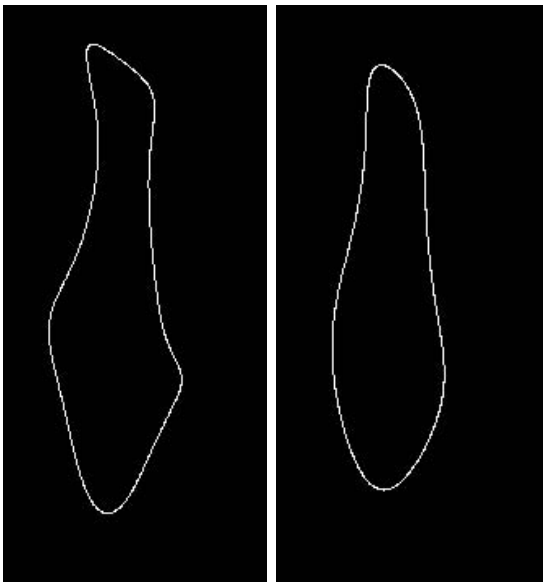

Figure 19: Left: The fish contour blurred by 2000 iterations. Right: The same contour after 6000 iterations

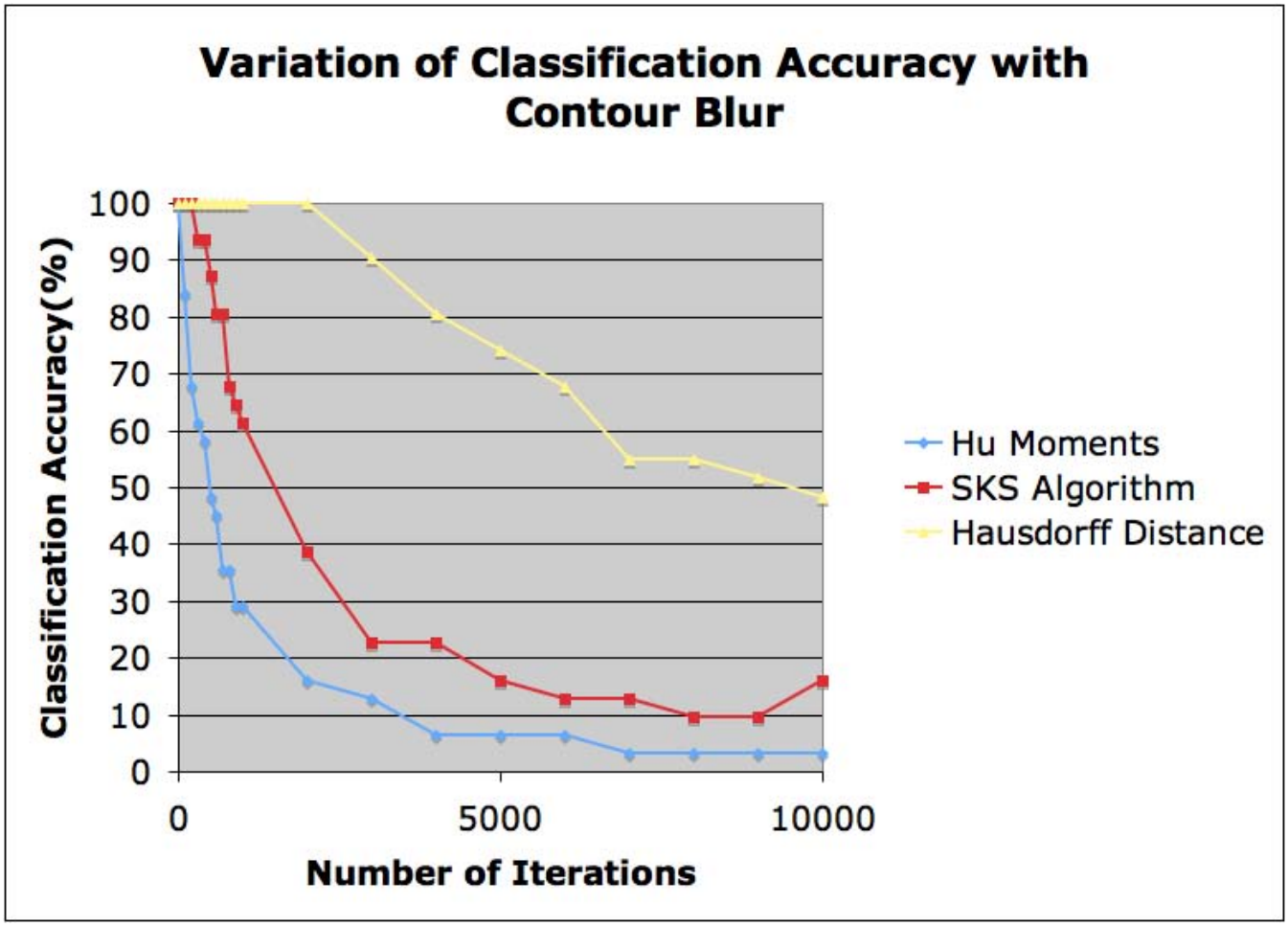

Figure 20: Performance of the SKS algorithm with a maximum-likelihood classifier using the Hu invariant moments.

6 Biological Computation

In this section, the components of the algorithm are discussed and architectures are proposed for performing the requisite computations. We specifically seek architectures which are plausable biologically.

The philosophy behind the SKS algorithm, developed in section 3, is heavily influenced by the working of the human visual system. In addition to developing a robust and invariant representation for shapes, this project seeks an explanation of how humans recognize shapes. One of the significant aspects of the SKS algorithm is that it is highly parallel. All the elements of the algorithm can be implemented on a parallel architecture like a neural network. The curvature computation algorithm is local to the object boundary and, except for second-order noise effects, independent of orientation. The feature vector computed for each point is also independent of the process of calculation of feature vectors at other locations.

In this section a neural network architecture that implements the algorithm, developed and tested in the previous sections, is discussed. First a summary of the visual pathway and the processing of shapes in the human visual cortex is presented. This is followed by a discussion of some neural network architectures that are present in literature. *The Standard Model*[26][27][30][5][29], an architecture for shape processing developed at the Computer Science and Artificial Intelligence Laboratory, MIT and the "VisNet," proposed by Rolls and Deco in [28] are discussed here. Both are hierarchical neural network implementations of the visual pathway. They achieve scale and translational invariance and closely mimic the behavior of actual cortical recordings of cells in the higher visual areas (V4 and IT). Finally, an augmented model that ensures invariance to rotation and brings about an object centered representation of shapes is proposed.

6.1 The Visual Pathway

The processing of shape information starts very early in the visual pathway (figure 21). The receptors (cones in this presentation of the mode) on the retina are packed densely in a region called the fovea. This region is normally also focus of attention. The receptors are connected to the bipolar cells in the retina either directly (in an excitatory role) or through horizontal cells (in an inhibitory role)[28]. This configuration generate center surround ON and OFF receptive fields. A detailed description of this architecture and the processing is present in [17][28]. The bipolar cells feed into the retinal ganglion cells which in turn feed through the optic nerve onto the *lateral geniculate nuclei*(LGN).

Retinotopic mapping, in which the geometric positioning of cells corresponds to light sensors in the retina, is maintained through the LGN to higher visual areas. The LGN contains two kinds of cells - Magno-cells (M-LGN) and Parvo-cells (P-LGN). Of these cells, the P-LGN are primarily responsible for communicating shape information. Both these layers of the LGN feed into different regions of the primary visual cortex (V1). Since the focus of this description is on shape processing, only the section of the visual cortex relevant to shape is discussed from here.

6.1.1 V1

The primary visual cortex constitutes several layers (hence the name *striate cortex*). There are rich vertical connections between these layers with very little horizontal or diagonal spreading between layers. This would suggest that most of the processing in the V1 area is local and it clearly is not the *seat of perception* [17]. There are three types of cells in this visual area. *Simple Cells*

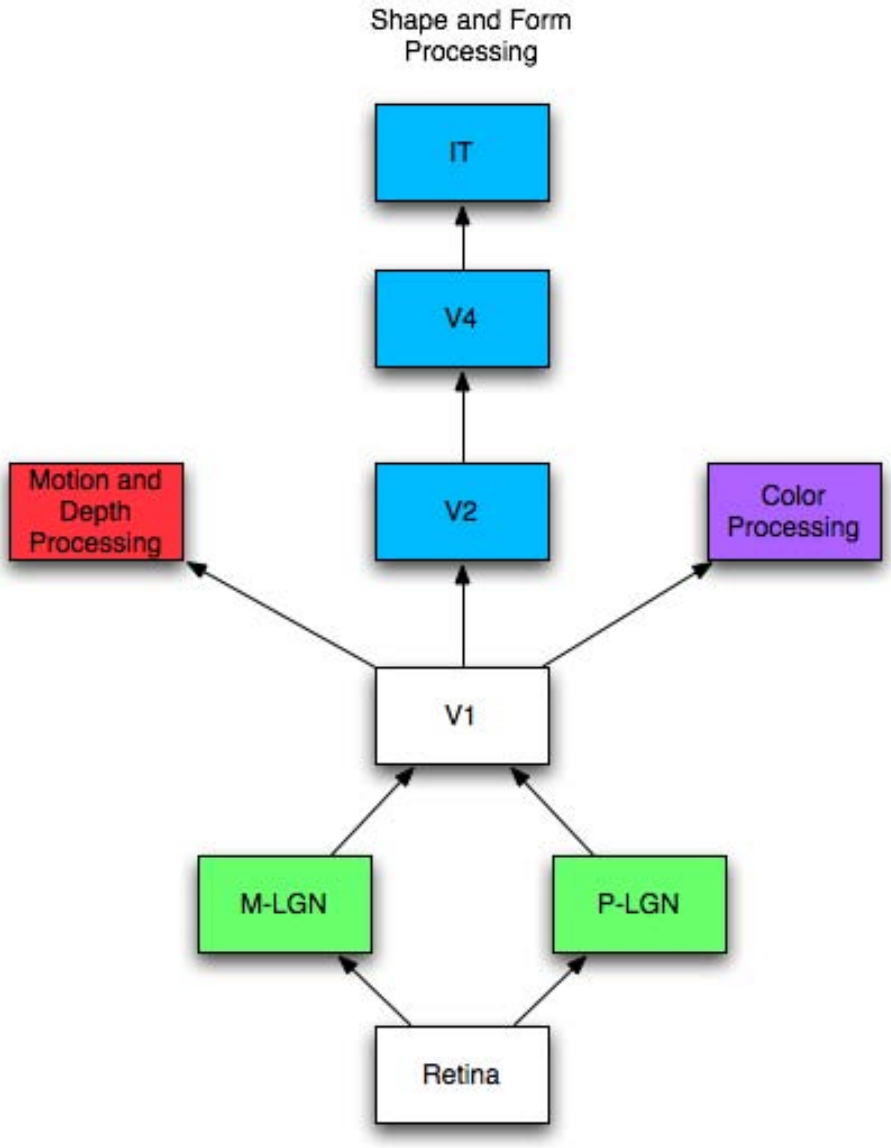

Figure 21: A block diagram of the Visual Pathway.

respond best to bars or edges passing through the center of the receptive field corresponding to that cell, and having specific orientations. They are tuned to four different orientations with a tuning window of 45^o [28]. They are said to arise out of the spatial summation of ON and OFF center cells[17][28]. Research by De Valois and De Valois also show that these cells are sensitive to the spatial frequency of the visual stimuli[28]. These properties of the Simple Cells are essential to understanding their implementation using Gabor spatial filters in later sections. *Complex Cells* are also sensitive to oriented bars but they show some translational invariance. The bars can be present anywhere in their receptive field. Hubel [17] proposes that this response of complex cells is due to the integration of the responses of simple cells with similar orientation tuning but different (adjacent) positions in the visual field. *End Stopped Cells* are the third type of cells present in the V1 area. They are sensitive to curves with specific curvatures and properly oriented lines of fixed lengths.

The cells in V1 are arranged in "hypercolums" [28]. These "hypercolumns" are responsible for the processing of visual information from one specific area in the visual field. They contain columns of neurons tuned to a particular orientation with multiple columns representing the four different tunings. This is important because the retinotopic mapping is still maintained and the different hypercolumns can be viewed as tiny processors of local shape information.

In [28] it is suggested that V1 layer neurons "may function collectively to incorporate contextual information from outside their classical receptive fields and in turn serve pre-attentive visual segmentation". This is important to the implementation of the neural network architecture for the SKS algorithm, because global shape information like the principal axis may require such mechanisms to enable its incorporation to the processing of contours.

6.1.2 Extra-striate Processing of Shape Information

There is very little understood about the behavior of brain circuits external to the striate cortex. However, some studies [29][23][24] have shown that these neurons are sensitive to combinations of features from their afferent layers. The extra-striate areas primarily involved in handling shape information are regions of the V2, V4 and the inferior temporal cortex (IT). The cells in the V2 are known to be sensitive to low order combinations of the afferent cells from the striate cortex (the primary visual cortex) [28][29]. These combinations represent the growing complexity of features along the visual hierarchy. They also represent the growing invariance of this architecture to small translations. This has been replicated by architectures like the VisNet [28] and *the Standard Model*[26]-[29].

The V4 is known to be sensitive to high order feature combinations [24][8]. There are cells in the V4 that are tuned to simple features like oriented line segments and others that are tuned to more complex curvatures. In [8] Connor summarizes his research into neurons in the V4. He shows that these neurons are sensitive to arrangement of shape features relative to the object center (which implies an object-centered coordinate system). The V4 neurons are shown to be sensitive to angular position of a feature, the normal at the feature, the curvature of the feature and the context (the curvatures of the segments adjacent to features). These cells also show some invariance to object size and object location (translation). They also have larger receptive fields when compared to the cells lower in the hierarchy. These properties are of significance to the the SKS algorithm and

also the neural network architecture. Connor [8] hypothesizes an object recognition framework that is a hierarchical feed-forward network with neurons sensitive to feature combinations of their afferent layers. This network not only encodes what features are present but also their relative spatial arrangements. This hypothesis along with the research of Poggio et al. [26]-[29] and Rolls' VisNet[28] were essential to the development of the neural network architecture to implement the SKS algorithm.

The final higher visual area that is significant to shape processing is the Inferior Temporal cortex (IT). The neurons in this area of the visual pathway are sensitive to moderately complex features [28]. Neurons that are sensitive to similar features are clustered together and are involved in competitive inhibition to ensure discrimination. These neurons also exhibit significant translational invariance [28] in addition to scale invariance of up to two octaves [29]. Poggio et al. [29] argue that this layer contains several view tuned neurons whose responses are pooled to produce view independent neurons that identify specific objects. Additionally, Rolls [28] has documented that this representation of objects in the IT is distributed as this enables the encoding of a significantly larger number of shapes. This also ensures a graceful degradation of performance in the presence of noise. A significant finding about the view independent cells in the IT [28] is that a set of neurons were shown to fire even when the object presented was completely inverted. This suggests an object based framework for recognition in these areas.

6.2 VisNet and the Standard Model

Several architectures for invariant object recognition are present in literature [29][28][14][4]. Of these neural network implementations, those of Poggio et al. [29] and Rolls [28] present a biologically plausible framework for processing shapes (2D and 3D). In this section these two models of human vision are presented and discussed. Both are feature hierarchies and exhibit strong invariance to translation and scale changes. Both are intent on explaining the processing of shapes in the visual pathway using biologically plausible circuits. They both are feed-forward networks with absolutely no top-down feedback.

6.2.1 VisNet

VisNet is a neural network architecture for shape recognition developed by Rolls and Deco and described in their book [28]. The general philosophy involves a feature hierarchy going from simple (oriented lines) to complex (curves and corners) features. The network has 4 layers of neurons that learn and classify using mutual inhibition (over short ranges) and competition. The units from one layer converge onto neurons in the higher layers. In other words several neurons in the lower layers are afferent on neurons in the higher layers. This implies an increasing size of receptive fields as we go higher up in the network. The input to the network is from a set of 2D spatial filters implemented as "Difference of Gaussians" (DOG). These are intended to mimic the orientation and spatial frequency sensitivities of *simple* cells in V1. Rolls and Deco do not assume any preexisting affinity for any combination of features. The network learns through self-organization to represent the entire feature space. Feature combinations are not replicated at all positions. Instead a representative sample of images with all possible features is learnt by the lower

layers of the network. These images are presented at all possible positions in the input layer. The higher layers learn feature associations specific to objects and do not bother with translational invariance.

Rolls and Deco also suggest a trace rule for learning. This is similar to Hebbian learning except there is a temporal aspect to it. This temporal aspect to learning brings about some of the invariance to scale and translation. This form of Hebbian-like learning also makes the network biologically plausible. Additionally, the competitive nature of learning also provides a distributed representation similar to that exhibited by neurons in the IT. Overall, this hierarchical representation exhibits the ability to differentiate between different spatial arrangements of features and is similar to the behavior of cells between the V1 and IT layers of the ventral pathway.

6.2.2 The Standard Model

The Standard Model [26][29] is a hierarchical model of shape recognition developed by Poggio et al. It is a strictly feed-forward architecture and seeks to explain the first 150 ms of vision. It is similar to the hierarchical model proposed by Fukushima [14] in that there are alternating layers of *simple* and *complex* cells that learn feature associations. In the Standard Model, these two types of cells are responsible for two distinct tasks, but unlike the VisNet, their roles are predefined and do not emerge out of self-organization. The simple cells (S) are responsible for template matching. Each cell is active when the features in the object at a particular spatial location have high correlation with the feature that the S cell is tuned for. At the very base the S cells mimic the behavior of the *simple* cells in V1. They are sensitive to oriented line segments. The entire space of orientations is covered by cells with a tuning peaks at 45^o intervals. In the Standard Model 2D Gabor filters serve as inputs to this layer of S-cells. These cells are sensitive to location of the oriented bars and can be assumed to be part of the "hypercolumns" that are found in V1. The *complex* cells (C) are responsible for the invariance characteristics of the network. They achieve this by pooling in inputs from S-cells that are tuned to the same feature but located at slightly different positions in the visual field. This operation is achieved by a softmax of inputs from the S-cells layer.

These layers of C and S cells are alternated to produce features of increasing complexity while ensuring a small degree of invariance to scale and translation. A complete discussion of this model is presented in [29]. In the same paper the authors discuss the biological plausibility of these operations of shape tuning and softmax. They argue convincingly that circuits for these operations are biologically plausible. This hierarchy of alternating layers agrees with neurophysiological data in literature. Most significantly, the layers at the top of the hierarchy (C2) show considerably similar sensitivities as cells in area V4 as reported in [23][24]. The authors state [5] that this hierarchy results in an object centered representation of features as hypothesized by Connor [8]. The results presented confirm this claim.

7 The SKS Neural Network Architecture

The neural network architecture proposed in this section is influenced by VisNet and the Standard Model. The philosophy of the SKS algorithm is to utilize global characteristics of a shape to over come rigid similarity transforms and use local features to arrive at a shape description. The SKS algorithm brings about significant invariance to rotation, scale and translation. In addition it also exhibits robustness to noise and blur. Translating the serial computations of the algorithm discussed in section 3 to a parallel architecture like a neural network is simplified by the fact that the modules of the algorithm are inherently parallel. The DSS algorithm for calculating curvature is also remarkably parallel. In this section the specifics of the architecture to implement the SKS algorithm in a neural network are discussed. No simulations of the proposed neural network were carried out.

7.1 Description of the Architecture

The SKS architecture inherits some prominent features from the VisNet and the Standard Model. In addition to the "what" information that is processed it also seeks to incorporate an object-based frame of reference to arrive at a consistent and invariant representation of shape. The Standard Model achieved scale and translation invariance by utilizing tuning functions and the pooling outputs over spatial location and scale. Rotational invariance is not discussed. Different view-tuned neurons are present for different views of the object and intermediate views are determined by graded response of these neurons.

In our hypothesis however, it is proposed that view-tuned neurons exist but they are for depth-rotated analogues. The presence of different neurons for objects rotated in the plane is inefficient because the same set of features are "visible" to the shape processors. Instead, we observe that humans require longer to process rotated images and make more errors in recognition, and this degredation in performance is a monotonic function of rotation angle[18]. With this observation, we propose that the parallel, high speed, "wired" sort of processing is minimally rotation invariant. It is possible to propose a "biologically plausible" architecture that could implement this kind of processing. The hypothesis is influenced by the fact that vision involves the perception of "orientation" of an object (usually in terms of the principal axis or the axis of elongation) in addition to its spatial location (in terms of its geometric center).

The neural network architecture hypothesized has the following features (figure 22):

- The architecture is a hierarchical, feed-forward, multilayer, competitive neural network with mutual inhibition within a layer. The alternating simple(S) and complex(C) layers design of the Standard Model is retained. In essence it is a feature hierarchy going from units sensitive to oriented edges to complex stimuli.
- The input to the network is a 2D Gabor filtered image, simulating the behavior of orientation-sensitive retinal receptive fields. The filtering of images using 2D Gabor spatial filters is discussed in [9].

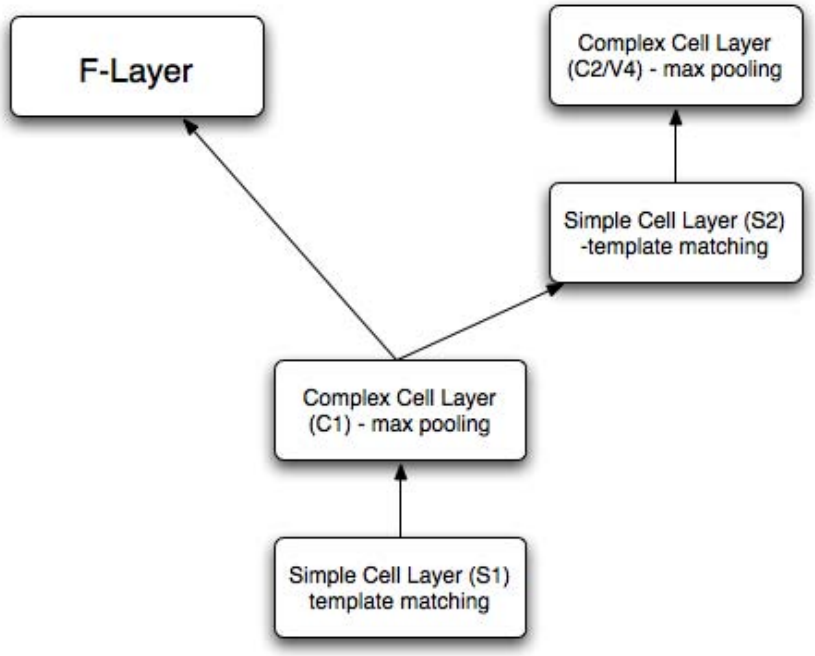

Figure 22: The SKS Neural Network Architecture. Alternating layers of Simple and Complex cells produce cells that are sensitive to low order combinations of features and at the same time a degree of translational and scale invariance. Additional layers can be added after C2 in order to arrive at more complicated features and larger receptive field sizes.

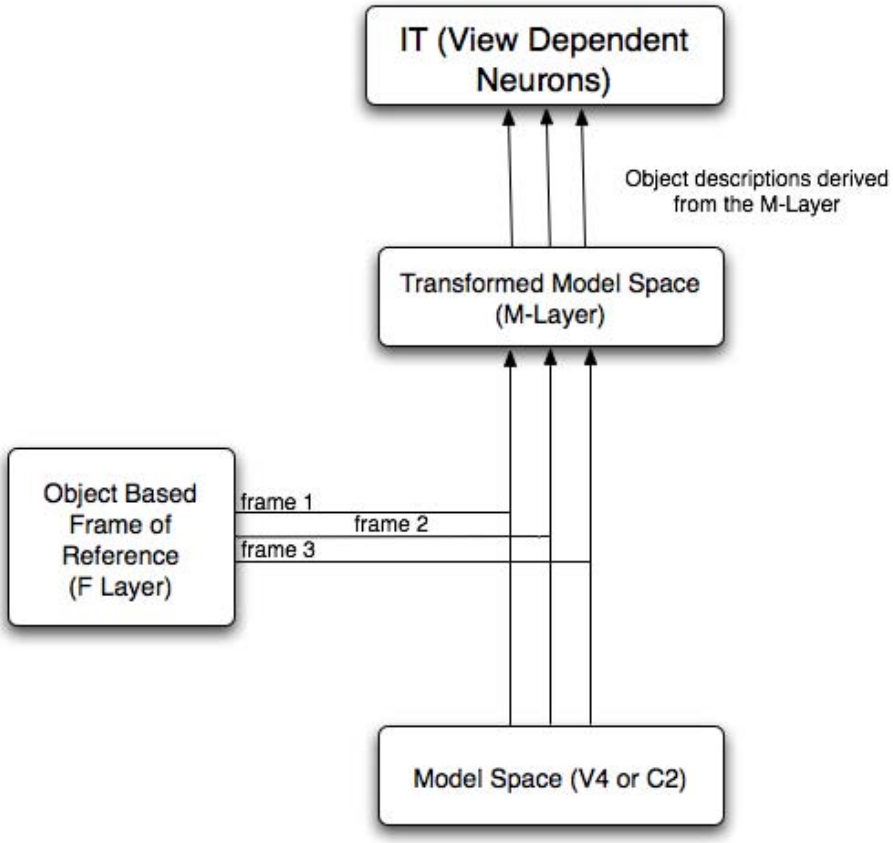

Figure 23: Sigma Pi connections gate the connections between the C2 layer and the M-Layer. Each cell in the M-layer is connected to equivalent cells in the C2 layers. When an afferent cell fires the input is communicated to the cell in the M-Layer that is consistent with that assignment of principal axis.

- The architecture presented hypothesizes that the feature-specific information and the "global" information with regard to a shape are processed in parallel. After one layer of alternating S and C cells, the output of the C layer branches out into two distinct branches. One branch is for processing the orientation center of the shape (F layer). The second branch continues the process of combining the outputs of layer C1 to produce low order feature combinations in layer S2.
- The C2 layer is equivalent to cells in the V4. This in turn is equivalent to the "model" space discussed in section 3. The cells are sensitive to angular position and curvature (added to distance) relative to the object center. However, the angular position here is relative to the viewer coordinates.
- The M-Layer, in figure 23, is a transform space of the C2 layer. Considering the C2 layer to be parameterized by distance, angular position and curvature of a segment [23][24][8], the M-Layer is similarly parameterized. Each cell in C2 tuned to a particular feature at particular distance from the object center is connected to each and every cell tuned to the same feature at the same distance from the object center (but at different angular positions) in the M-Layer. These connections are gated by the outputs of the F-layer. For a particular orientation of an object a combination of the firing of cells in the F-layer and the C2 layer will produce (related) activity in the M-Layer.
- The M-Layer then communicates by simple feature combination a view (rotation in depth) dependent representation to the IT.

The critical additions to the architecture are the M-layer and the F-layer. Hinton in [15] recommends an architecture to enable invariance to similarity transforms by modulating shape descriptions by object-based frames of reference.The F-layer computes an object-based frame of reference relative to which the object description is communicated to the IT (through the M-layer). The significance of the F layer is shown in figure 23. The F-layer outputs gate the neural connections between the V4 (C2) layer and the M-layer. The gating enables only consistent object descriptions to pass through to the cells in the IT. Object descriptions in which there is correlation between the principal axis and the arrangement of features (segments of different curvatures) relative to this framework are considered to be consistent. This gating is accomplished by "biologically plausible" sigma-pi connections between the two sets of neural circuits [28]. The M-layer is a transform space where all features are transformed relative to a reference frame defined by the principal axis. Each view dependent neuron is activated by an object description corresponding to combinations of features from the M-layer. The outputs of the view-dependent neurons are pooled for the activation of view-independent neurons in higher areas of the IT.

7.2 Equivalence to the SKS algorithm

There is considerable similarity between the SKS algorithm and the neural network described above. There are parallels that can been drawn between the modules of the SKS algorithm and this neural network. The two are philosophically equivalent.

In model building using the SKS algorithm, the first steps of processing include determining the

geometric center of the contour and, in some implementations, the principal axis. This is accomplished in part by the alternating S and C layers of the hierarchy, as they achieve a degree of translational and scale invariance. The determination of the principal axis is done by the F-layer. The inputs to this layer are the outputs of the C1 layer. The F-layer is modeled as a densely connected layer of neurons in which all the feature specific cells of C1 are connected to N neurons corresponding to N finely tuned orientation tuning functions spanning the range of orientations of the principal axis. The F-layer connections can be assumed to be hard wired as of now. For each contour a sparse firing of the N neurons of the F-layer indicates a principal axis of a specific orientation. The outputs of these N neurons are used to gate the connections between the C2 layer and the M-Layer. In the SKS algorithm the contour is transformed to the new frame of reference defined by the principal axis. In the network the model space equivalent of the contour is transformed instead. These two are equivalent operations. An assumption made here is that the gating circuits and the F-layer circuits are hard-wired (either genetically or during development).

In the SKS algorithm, curvature is determined utilizing the digital straight segments (DSS) algorithm. This algorithm can be implemented in a parallel architecture. The functioning of the feature hierarchy can be viewed as curvature filtering [30][29]. Along the hierarchy low order combinations of afferent features produces sensitivity to more complex stimuli (like curves of varying curvature). In the SKS algorithm each and every point is treated uniquely and technically an infinity of "features" (different curvatures) is present. In the network however, these combinations are dictated by visual experience and the statistics of real images. Additionally, graded response of different features effectively provides a continuous feature space. The model space in the SKS algorithm is similar to the C2 layer which is parameterized by curvature of the feature, angular position and distance from the object center. The significant difference is in the matching process. Although there is no explicit accumulator present in the network discussed above, the accumulator can be viewed as 2D spatial correlation between the models of two contours. This correlation is implemented in the connections between the neurons of the M-Layer and the view-tuned neurons of the IT. If no depth rotation is assumed, these view-tuned cells should be sufficient to represent and classify the database of shapes presented in section 5.

In this section, a neural network architecture for shape recognition is proposed. It is largely derivative from models developed by Poggio and Riesenhuber , and Rolls and Deco. The models have been augmented with additional layers (F-layer and M-layers) that enables a rotationally invariant representation of shapes. There are distinct representations for depth rotated views as the feature sets that are visible are not only different but they also have different spatial arrangements. The final neural network developed is influenced by the SKS algorithm and is philosophically similar to it. Implementation of this network is beyond the scope of this report.

8 Conclusion and Extensions

In this project, we have explored a new approach to two-dimensional shape recognition. The method draws from literature on the Hough transform and its extensions. The methods is shown to be invariant to zoom, translation, rotation, and partial occlusion, although not all simultaneously. The method is shown to be robust to distortions which smooth the contour shape. Furthermore, when the method misclassifies a shape, it chooses a shape which is most "similar" (in a human-intuitive sense) to the original.

The computer-based version of the algorithm has been shown to have a reasonable implementation in neural hardware.

We need to finish the comparison with classifiers based on invariant moments. This work is under way at this time.

There is a great deal more to be done.

The neural network version of the algorithm needs to be simulated carefully, and its performance evaluated.

The current implementation of the algorithm is completely invariant to rotation, but this is not necessary. Observations of humans performing shape recognition tasks suggest that rotational invariance may be restricted to a few degrees, but invariance to partial occlusion may be more important. There are probably ways to accomplish this. For example, Edelman and Poggio [13] show that models can be developed which do not depend on the center of gravity but on a collection of reference points, where each reference point is expressed relative to other reference points. Such a philosophy may allow simultaneous zoom and partial occlusion.

We know how the lower levels of the human visual system work: the behavior of retinal and higher-level receptive fields. The SKS algorithm could be developed in terms of the outputs of such fields, rather than the current point-based approach, probably with considerable gain in speed performance.

References

[1] M. J. Atallah. A linear time algorithm for the hausdorff distance between convex polygons. *Information Processing Letters*, 17, 1983.

[2] D. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2), 1981.

[3] E. Belogay, C. Cabrelli, U. Molter, and R. Shonkwiler. Calculating the hausdorff distance between curves. *Information Processing Letters*, 64, 1997.

[4] Irving Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[5] C. Cadieu, M. Kouh, M. Riesenhuber, and T. Poggio. Shape representation in v4: Investigating position-specific tuning for boundary conformation with the standard model of object recognition. Technical Report CBCL Paper 241/AI Memo 2004-024, Massachusetts Institute of Technology, 2004.

[6] David Coeurjolly, Serge Miguet, and Laure Tougne. Discrete curvature based on osculating circle estimation. *Lecture Notes in Computer Science*, 2059:303–, 2001.

[7] David Coeurjolly and Stina Svensson. Estimation of curvature along curves with application to fibres in 3d images of paper. *Lecture Notes in Computer Science*, 2749:247–254, 2003.

[8] Charles E. Connor. Shape dimensions and object primitives. *The Visual Neurosciences*, 2:1080–1089, 2003.

[9] John G. Daugman. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression complete discrete 2-d gabor transforms by neural networks for image analysis and compression complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 36(7), July 1988.

[10] FranÃ§ois de Vieilleville, Jacques-Olivier Lachaud, and Fabien Feschet. Maximal digital straight segments and convergence of discrete geometric estimators. *Lecture Notes in Computer Science*, 3540:988–997, 2005.

[11] S.R. Deans. Hough transform from the radon transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(2), 185188, March 1981.

[12] J.-P. Debled-Rennesson, I. Reveilles. A linear algorithm for segmentation of digital curves. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(4):635–662, 1995.

[13] S. Edelman and T. Poggio. Bringing the grandmother back into the picture: A memory-based view of object recognition. *MIT AI Memo*, (1181), 1991.

[14] K. Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988.

[15] G. E. Hinton. Shape representation in parallel systems. In *Proceedings of Seventh International Joint Conference on Artificial Intelligence*, volume 2, pages 1088–1096, Vancouver, BC, Canada, 1981.

[16] P.V.C. Hough. Method and means for recognizing complex patterns. *U.S. Patent*, 3069654, 1962.

[17] David H. Hubel. *Eye, Brain and Vision*. W.H. Freeman and Company, 1995.

[18] P. Jolicoeur. The time to name disoriented natural objects. *Memory and Cognition*, 13, 1985.

[19] Reinhard Klette and Azriel Rosenfeld. *Digital Geometry: Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann, 2004.

[20] V.A. Kovalevsky. New definition and fast recognition of digital straight segments and arcs. *Pattern Recognition, 1990. Proceedings., 10th International Conference on*, 2:31–34, 16-21 Jun 1990.

[21] M.Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8, 1962.

[22] E. Oja, J. Karhunen, L. Wang, and R. Vigario. Principal and independent components in neural networksrecent developments, 1995.

[23] A. Pasupathy and C.E. Connor. Responses to contour features in macaque area v4. *Journal of Neurophysiology*, 82:2490–2502, 1999.

[24] A. Pasupathy and C.E. Connor. Shape representation in area v4: Position-specific tuning for boundary conformation. *Journal of Neurophysiology*, 86:2505–2519, 2001.

[25] A. Pasupathy and C.E Connor. Population coding of shape in area v4. *Nature Neuroscience*, 5:1332–1338, 2002.

[26] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, pages 1019–1025, 1999.

[27] M. Riesenhuber and T. Poggio. *The Visual Neurosciences*, volume 2, chapter How visual cortex recognizes objects: The tale of the standard model, pages 1640–1653. The MIT Press, 2003.

[28] Edmund Rolls and Gustavo Deco. *Computational Neuroscience of Vision*. Oxford University Press, 2001.

[29] T. Serre, M. Kouh, C. Cadieu, U. Knoblich, G. Kreiman, and T. Poggio. A theory of object recognition: computations and circuits in the feedforward path of the ventral stream in primate visual cortex. Technical Report CBCL Paper 259/AI Memo 2005-036, Massachusetts Institute of Technology, 2005.

[30] T. Serre, L. Wolf, and T. Poggio. A new biologically motivated framework for robust object recognition. *MIT AIM*, 2004.

[31] W.E. Snyder and H. Qi. *Machine Vision*. Cambridge University Press, 2004.

[32] Anne Vialard. Geometrical parameters extraction from discrete paths. In *DCGA '96: Proceedings of the 6th International Workshop on Discrete Geometry for Computer Imagery*, pages 24–35, London, UK, 1996. Springer-Verlag.

[33] Marcel Worring and Arnold W. M. Smeulders. Digital curvature estimation. *CVGIP: Image Underst.*, 58(3):366–382, 1993.

[34] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37, 2004.